16時より開始いたします。もうしばらくお待ちください

- 後日ご共有の為、本セミナーは録画させていただきます。
- SmartDB環境にログインしてお待ちください。

  ※FirefoxまたはchromeでのZoomの実施をお願いいたします。
- Zoomの操作についてはチャット機能でご質問ください。

  送信先を「全員」にした状態でチャットしてください。

  ※宛先が全員でない場合投稿に気づけない場合がございます。

- 本日使用するPower Shellコードのご確認をお願いします。

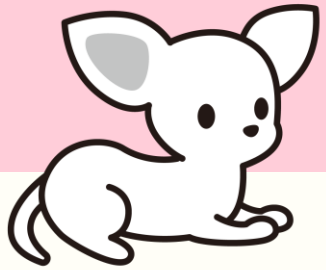
  【3/8】に送付している案内メールの添付資料に記載しております。

15:00より開始いたします。もうしばらくお待ちください。

スマラジ！ SmartDB Radio

活用レベルアップ！

SmartDB深掘りセミナー



スマラジ！SmartDBのAPIを使ってみよう！
～全6回で習得！REST APIアプリ操作基本編②～

株式会社ドリーム・アーツ

協創パートナー推進本部
中川 嵩之

DreamArts

1. 前回の復習
2. SmartDB REST APIの始め方
3. よく使うSmartDB REST API
4. ハンズオン
5. まとめ
6. 次回予告

DreamArts

1. 前回の復習
2. SmartDB REST APIの始め方
3. よく使うSmartDB REST API
4. ハンズオン
5. まとめ
6. 次回予告

1. APIとは、外部から**機能・情報を利用するための窓口**である
2. APIには**リクエスト**と**レスポンス**という2つの通信がある
3. REST APIでは、データの**取得**、**登録**、**更新**、**削除**ができる
4. リクエストは、主に4つの要素で構成される

要素	説明
URI	：処理を行う対象のリソース
HTTPメソッド	：実行する処理内容
リクエストボディ	：登録、更新するデータの詳細
リクエストヘッダ	：認証などの補足情報

5. レスポンスは、主に2つの要素で構成される

要素	説明
ステータスコード	：実行した結果
レスポンスボディ	：返却されたデータの詳細

DreamArts

1. 前回の復習
2. SmartDB REST APIの始め方
3. よく使うSmartDB REST API
4. ハンズオン
5. まとめ
6. 次回予告

STEP 01 バインダトークンの契約

STEP 02 バインダトークンの発行

STEP 03 バインダトークンの連携

STEP 04 実行プログラムの開発

バインダAPIトークンのオプションを契約

オプション	価格
バインダ追加	月額 100,000円/20バインダ
コラボレーター機能	月額 3,000円/10コラボレーター
バインダAPI	**月額 100,000円/10トークン**
ディスク容量追加	営業担当までご相談ください
ファイル一括アップロード	1,200,000円/1ライセンス ※購入から1年間の利用権利、期間終了時に失効

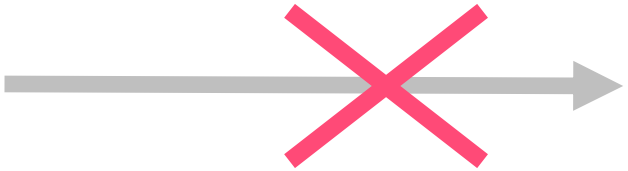
DreamArts

STEP 01 バインダトークンの契約

STEP 02 バインダトークンの発行

STEP 03 バインダトークンの連携

STEP 04 実行プログラムの開発



スマートデービー SmartDB

第三者からの不正利用を防止するなど
APIの安全性を保つためにトークンを利用

DreamArts

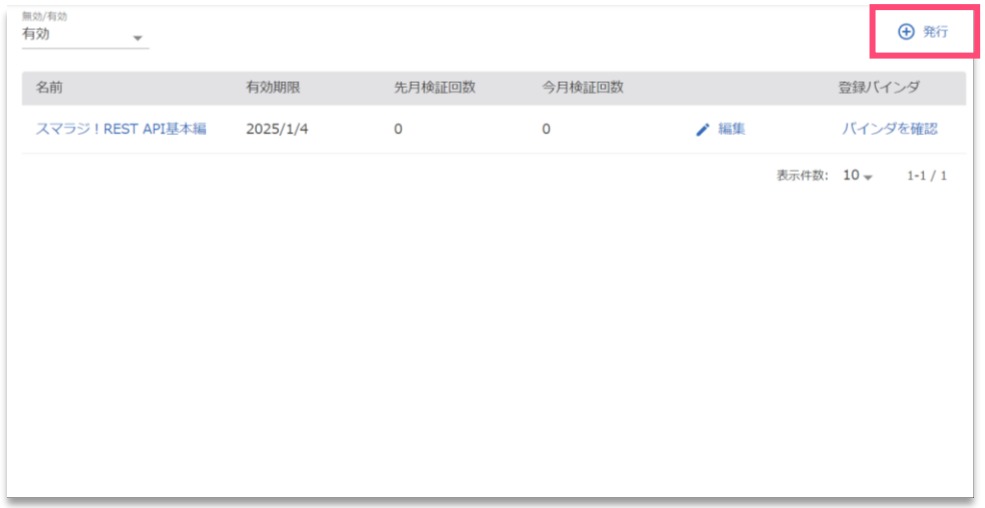
STEP 01 バインダトークンの契約

STEP 02 バインダトークンの発行

STEP 03 バインダトークンの連携

STEP 04 実行プログラムの開発

システム管理画面からバインダトークンを発行



システム管理＞認証とセキュリティ＞バインダトークン

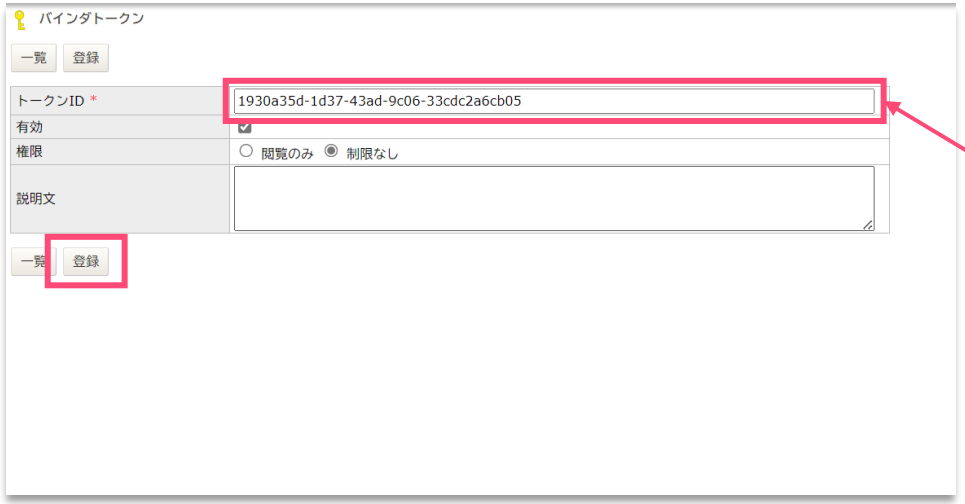
STEP 01 バインダトークンの契約

STEP 02 バインダトークンの発行

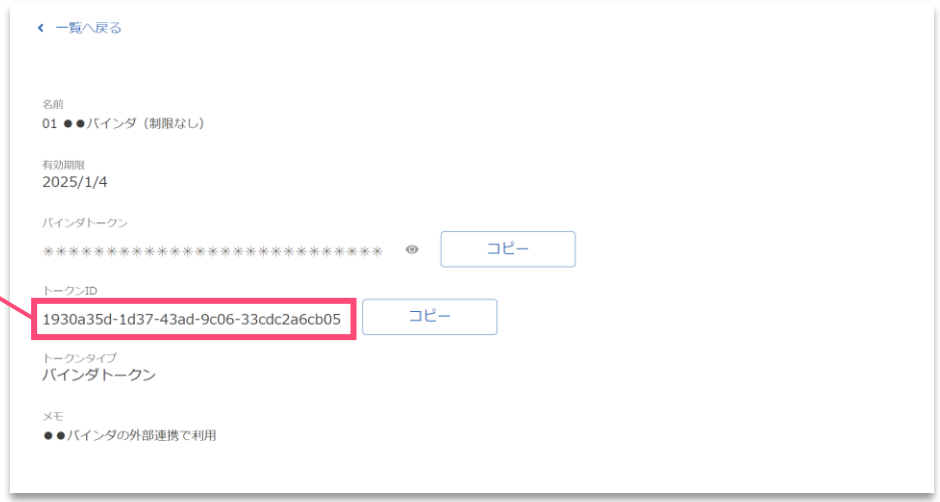
STEP 03 バインダトークンの連携

STEP 04 実行プログラムの開発

対象のバインダに**バインダトークンを紐づけ**



管理者用サイドメニュー > 外部連携 > トークン設定



システム管理 > 認証とセキュリティ > バインダトークン

DreamArts

STEP 01 バインダトークンの契約

STEP 02 バインダトークンの発行

STEP 03 バインダトークンの連携

STEP 04 実行プログラムの開発

SmartDBの情報・RestDocを参照して**リクエストを作成**

参照情報

リクエスト

- URI
- HTTPメソッド
- リクエストヘッダ
- リクエストボディ

問題

SmartDBのREST API利用の流れとして正しいのは？

※ DCS / オンプレミスの場合、手順が異なる場合がございます。

1. SmartDB REST APIではトークンを利用する必要がない。
2. SmartDB REST APIでは第三者の不正利用を防止するためにトークンを利用する必要がある。
3. バインダトークンを発行後、業務プロセス定義とトークンを連携させる必要がある。
4. SmartDB REST APIのリクエストを作成するためにすべてのAPIを暗記する必要がある。

DreamArts

問題

SmartDBのREST API利用の流れとして正しいのは？

※ DCS / オンプレミスの場合、手順が異なる場合がございます。

1. SmartDB REST APIではトークンを利用する必要がない。
2. **SmartDB REST APIでは第三者の不正利用を防止するためにトークンを利用する必要がある。**
3. バインダトークンを発行後、業務プロセス定義とトークンを連携させる必要がある。

解説

1. SmartDB REST APIを利用する場合、セキュリティ向上のために**バインダトークン**が必要である
2. バインダトークンは、REST APIを利用する**バインダと連携する**必要がある
3. リクエストは**SmartDB環境情報**や公開ドキュメントである**RetsDoc**を参照して作成できる

DreamArts

1. 前回の復習
2. SmartDB REST APIの始め方
3. よく使うSmartDB REST API
4. ハンズオン
5. まとめ
6. 次回予告

SmartDB REST APIの一覧

BINDER
- バインダー覧取得
- バインダ詳細情報取得
- 部品定義情報取得
- よく使うバインダー覧取得
- よく使うバインダー覧に追加
- よく使うバインダー覧から削除
- ビュー一覧取得
- ビュー詳細情報取得

DOCUMENT
- 文書詳細情報取得
- 文書新規登録
- 文書更新
- 文書削除
- リスト型部品一覧取得
- リスト型部品削除
- リスト型部品の行新規登録
- リスト型部品の行更新
- リスト型部品の行削除
- ブランチ文書取得
- 自動再計算の履歴リスト取得
- 自動再計算の履歴取得
- 指定したビューの文書一覧取得
- 指定したビューの文書詳細情報取得
- 部品値分布取得
- 文書一覧のCSV出力
- CSV入力（新規）
- CSV入力（更新）
- コンプレックスCSV入力（新規）
- リッチコメントリスト取得
- リッチコメントの投稿
- リッチコメントの詳細情報
- リッチコメントの更新
- リッチコメントの削除
- リッチコメントのリアクション情報取得
- リッチコメントのリアクション更新
- 文書一括操作
- 文書既読・未読ユーザ数取得
- 文書既読情報取得
- 文書未読情報取得
- 文書既読・未読フラグ取得
- 文書既読フラグ設定
- バックグラウンドジョブステータス取得

PROCESS
- 開始可能業務一覧取得
- よく使う業務一覧取得
- よく使う業務一覧に追加
- よく使う業務一覧から削除
- 業務プロセス開始
- 開始済み一覧取得
- 実施済一覧取得
- 履歴一覧取得
- ワークリスト取得
- 実施履歴取得
- アクティビティ詳細情報取得
- アクティビティ実施
- 文書情報に基づいて実行中アクティビティ情報取得
- 指定ユーザのすべての代行設定取得
- 代行設定新規作成
- 代行設定取得
- 代行設定更新
- 代行設定削除

・
・
・

SmartDB REST APIの一覧

BINDER
- バインダー一覧取得
- バインダ詳細情報取得
- 部品定義情報取得
- よく使うバインダー一覧取得
- よく使うバインダー一覧に追加
- よく使うバインダー一覧から削除
- ビュー一覧取得
- ビュー詳細情報取得

DOCUMENT
- 文書詳細情報取得
- 文書新規登録
- 文書更新
- 文書削除
- リスト型
- リスト型
- リスト型部品
- 規登録
- リスト
- リス
- ブ
- 自
- 得

自動再計算の履歴取得
指定したビューの文書一覧取得
指定したビューの文書詳細情報取得

リッチコメントの削除
リッチコメントのリアクション情報取得
リッチコメントのリアクション更新
文書一括操作
文書既読・未読ユーザ数取得
文書既読情報取得
文書未読情報取得
文書既読・未読フラグ取得
文書既読フラグ設定
バックグラウンドジョブステータス取得

PROCESS
- 開始可能業務一覧取得
- よく使う業務一覧取得
- に追加
- から削除
- 細情報取得
- アクティビティ実施
- 文書情報に基づいて実行中アクティビティ情報取得
- 指定ユーザのすべての代行設定取得
- 代行設定新規作成
- 代行設定取得
- 代行設定更新
- 代行設定削除

SmartDBのREST APIってたくさん種類がある・・・。

何をどんな風に使えばいいんだろう・・・？

DreamArts

1. 前回の復習
2. SmartDBのAPIの始め方
3. よく使うSmartDB REST API
   - バインダ / 文書
   - 業務プロセス
4. ハンズオン
5. まとめ
6. 次回予告

BINDER

- バインダー覧取得
- バインダ詳細情報取得
- 部品定義情報取得
- よく使うバインダー覧取得
- よく使うバインダー覧に追加
- よく使うバインダー覧から削除
- ビュー一覧取得
- ビュー詳細情報取得

DOCUMENT

- 文書詳細情報取得
- 文書新規登録
- 文書更新
- 文書削除
- リスト型部品一覧取得
- リスト型部品削除
- リスト型部品の行新規登録
- リスト型部品の行更新
- リスト型部品の行削除
- ブランチ文書取得
- 自動再計算の履歴リスト取得
- 自動再計算の履歴取得
- 指定したビューの文書一覧取得
- 指定したビューの文書詳細情報取得
- 部品値分布取得
- 文書一覧のCSV出力
- CSV入力（新規）
- CSV入力（更新）
- コンプレックスCSV入力（新規）
- リッチコメントリスト取得
- リッチコメントの投稿
- リッチコメントの詳細情報
- リッチコメントの更新
- リッチコメントの削除
- リッチコメントのリアクション情報取得
- リッチコメントのリアクション更新
- 文書一括操作
- 文書既読・未読ユーザ数取得
- 文書既読情報取得
- 文書未読情報取得
- 文書既読・未読フラグ取得
- 文書既読フラグ設定
- バックグラウンドジョブステータス取得

DreamArts

BINDER

- バインダー覧取得
- バインダ詳細情報取得
- 部品定義情報取得
- よく使うバインダー覧取得
- よく使うバインダー覧に追加
- よく使うバインダー覧から削除
- ビューー覧取得
- ビュー詳細情報取得

DOCUMENT

- **文書詳細情報取得**
- **文書新規登録**
- **文書更新**
- **文書削除**
- リスト型部品一覧取得
- リスト型部品削除
- リスト型部品の行新規登録
- リスト型部品の行更新
- リスト型部品の行削除
- ブランチ文書取得
- 自動再計算の履歴リスト取得
- 自動再計算の履歴取得
- **指定したビューの文書一覧取得**
- 指定したビューの文書詳細情報取得
- 部品値分布取得
- **文書一覧のCSV出力**
- **CSV入力（新規）**
- **CSV入力（更新）**
- コンプレックスCSV入力（新規）
- リッチコメントリスト取得
- リッチコメントの投稿
- リッチコメントの詳細情報
- リッチコメントの更新
- リッチコメントの削除
- リッチコメントのリアクション情報取得
- リッチコメントのリアクション更新
- **文書一括操作**
- 文書既読・未読ユーザ数取得
- 文書既読情報取得
- 文書未読情報取得
- 文書既読・未読フラグ取得
- 文書既読フラグ設定
- バックグラウンドジョブステータス取得

DreamArts

SmartDBの文書を他システムに連携

他システムのデータをSmartDBの文書に連携

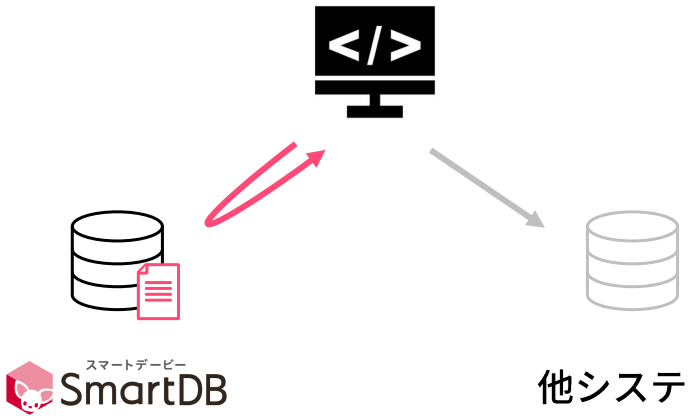
DreamArts

SmartDBの文書を他システムに連携

DreamArts

SmartDBの文書を他システムに連携

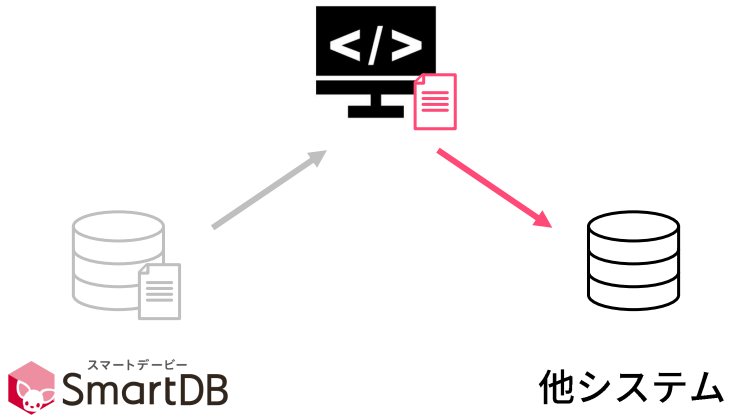
STEP 1



文書のデータを取得

文書詳細情報取得
指定したビューの文書一覧取得
文書一覧のCSV出力

STEP 2



他システムにデータを登録/更新/削除

他システムの提供するAPI

#	API	概要	利用シーン
1	文書詳細情報取得	指定された文書番号の文書データをJSONで取得 ※配置されたすべての部品データが取得されます	• 1回の処理で1つの文書データを処理 • SmartDBでの処理発生時に実行 例）申請書を連携
2	指定したビューの 文書一覧取得	指定されたビューで表示される文書一覧をJSONで取得 ※指定したビューIDの表示項目にある部品データが取得されます	• １回の処理で複数の文書データを処理 • 数十分、数時間おきに実行 例）承認済みの申請書をまとめて連携
3	文書一覧のCSV出力	指定されたビューで表示される文書一覧をCSVで取得 ※指定したビューIDの表示項目にある部品データが取得されます	• １回の処理で大量の文書データを処理 • 日時、週次で実行 例）マスタを連携

※業務要件などにより推奨されるAPIは異なる場合があり、実際にご利用する際は業務要件に適するAPIをご検討ください。

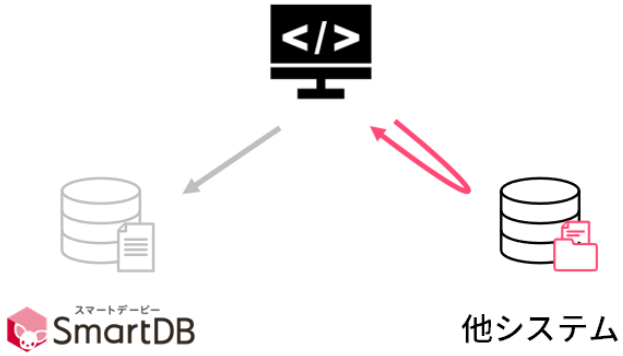
他システムのデータをSmartDBの文書に連携

他システムのデータをSmartDBの文書に連携

DreamArts

他システムのデータをSmartDBの文書に連携（文書新規登録）

STEP 1



他システムのデータを取得

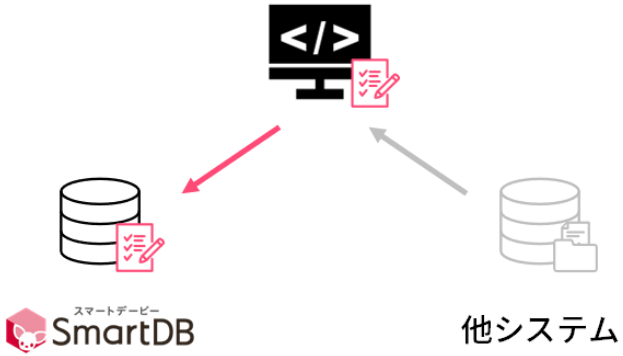
他システムの提供するAPI

STEP 2

対象文書の文書番号を取得

指定したビューの文書一覧取得

STEP 3



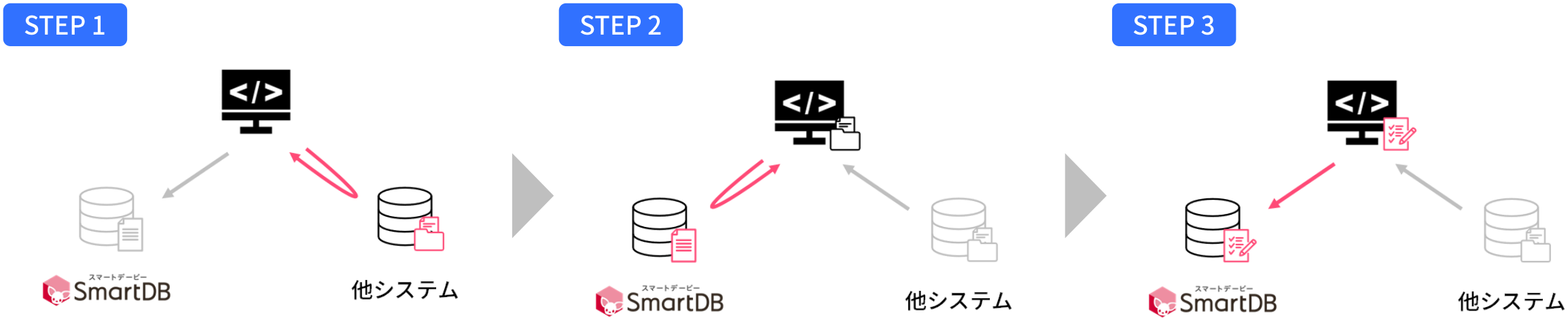
文書の新規登録/更新/削除

文書新規登録
文書一括操作
CSV入力（新規）

※STEP 3が「文書新規登録」の場合は不要

DreamArts

他システムのデータをSmartDBの文書に連携



他システムのデータを取得

他システムの提供するAPI

対象文書の文書番号を取得

指定したビューの文書一覧取得

文書の新規登録/更新/削除

文書更新 / 文書削除
文書一括操作
CSV入力（更新）

STEP 2「文書番号を取得」が必要な理由

元データ（外部システム）

項目	値
一意 法人番号	9011001040835
法人名	株式会社ドリーム・アーツ
担当者	◯山 ×太
	⋮

SmartDBの文書データ

項目	値
文書番号	6
一意 法人番号	9011001040835
法人名	株式会社ドリーム・アーツ
担当者	◯川 ×美
	⋮

1. REST API「文書更新」で変更対象の文書を指定するために、**文書番号を利用**している
   文書更新：https://{SmartDBのドメイン}.smartdb.jp/hibiki/rest/3/binders/{バインダID}/documents/**{文書番号}**
2. 外部システムでは**文書番号をデータとして持たない**（ケースが多い）
3. 複数のシステムで**一意の項目（法人番号、社員ID）を用いて**、文書番号を取得する

他システムのデータをSmartDBの文書に連携

DreamArts

#	API	概要	利用シーン
1	文書新規登録	指定された値で文書を登録 ※値を指定していない部品は設定された初期値で登録されます	• 1回の処理で1つの文書データを処理 • SmartDBでの処理発生時に実行 例）データを文書に連携
	文書更新	指定された文書番号の文書を更新 ※値を指定していない部品の値は更新されません	
	文書削除	指定された文書番号の文書を削除	
2	文書一括操作	複数文書の登録/更新/削除をまとめて実行	• １回の処理で複数の文書データを処理 • 数十分、数時間おきに実行 例）確定データをまとめて文書に連携 ※バインダが異なる複数の文書に処理を行うことも可能
3	CSV入力（新規）	CSVで文書を新規登録	• １回の処理で大量の文書データを処理 • 日時、週次で実行 例）マスタを連携 ※対象のバインダは１つのみ
	CSV入力（更新）	CSVで文書を更新	

※業務要件などにより推奨されるAPIは異なる場合があり、実際にご利用する際は業務要件に適するAPIをご検討ください。

問題

よく使うバインダ、文書のREST APIの説明として正しいのは？

※ DCS / オンプレミスの場合、手順が異なる場合がございます。

1. 文書の登録、更新、削除を行う場合、一度に扱う文書数に応じてREST APIを使い分ける。
2. SmartDBの文書データを他システムに連携することはできない。
3. 他システムのデータをSmartDBのバインダ及び文書に連携することはできない。
4. 更新・削除する文書は文書番号以外でも指定できる。

DreamArts

問題

よく使うバインダ、文書のREST APIの説明として正しいのは？

※ DCS / オンプレミスの場合、手順が異なる場合がございます。

1. **文書の登録、更新、削除を行う場合、一度に扱う文書数に応じてREST APIを使い分ける。**
2. SmartDBの文書データを他システムに連携することはできない。
3. 他システムのデータをSmartDBのバインダ及び文書に連携することはできない。

解説

1. 文書の登録、更新、削除のREST APIは複数あり、**文書数などに応じて**使い分ける
2. SmartDB REST APIを使えば、SmartDBと他システムを**双方向に連携**することができる
3. 文書の更新、削除を行う場合、**文書番号によって**対象文書を指定する必要がある。

DreamArts

1. 前回の復習
2. SmartDBのAPIの始め方
3. よく使うSmartDB REST API
   - バインダ / 文書
   - 業務プロセス
4. ハンズオン
5. まとめ
6. 次回予告

PROCESS

- 開始可能業務一覧取得
- よく使う業務一覧取得
- よく使う業務一覧に追加
- よく使う業務一覧から削除
- 業務プロセス開始
- 開始済み一覧取得
- 実施済一覧取得
- 履歴一覧取得
- ワークリスト取得
- 実施履歴取得
- アクティビティ詳細情報取得
- アクティビティ実施
- 文書情報に基づいて実行中アクティビティ情報取得
- 指定ユーザのすべての代行設定取得
- 代行設定新規作成
- 代行設定取得
- 代行設定更新
- 代行設定削除

よく使うSmartDB REST APIの一覧：業務プロセス

DreamArts

PROCESS

開始可能業務一覧取得
よく使う業務一覧取得
よく使う業務一覧に追加
よく使う業務一覧から削除
業務プロセス開始
開始済み一覧取得
実施済一覧取得
履歴一覧取得
ワークリスト取得
実施履歴取得
アクティビティ詳細情報取得
アクティビティ実施
文書情報に基づいて実行中アクティビティ情報取得
指定ユーザのすべての代行設定取得
代行設定新規作成
代行設定取得
代行設定更新
代行設定削除

PROCESS

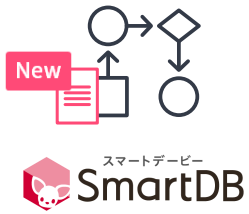
他システムの処理をトリガーに・・・

SmartDBで新規文書のWFを実行

他システム

○○という処理を実行

例）人事マスタデータを更新



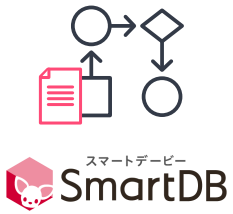
業務プロセスを開始

SmartDBで既存文書のWFを実行

他システム

○○という処理を実行

例）人事マスタデータを更新



業務プロセスを開始
アクティビティを実施

代行設定取得

DreamArts

PROCESS

他システムの処理をトリガーに・・・

SmartDBで新規文書のWFを実行

他システム

○○という処理を実行

例）人事マスタデータを更新

業務プロセスを開始

スマートデービー SmartDB

代行設定取得

SmartDBで既存文書のWFを実行

他システム

○○という処理を実行

例）人事マスタデータを更新

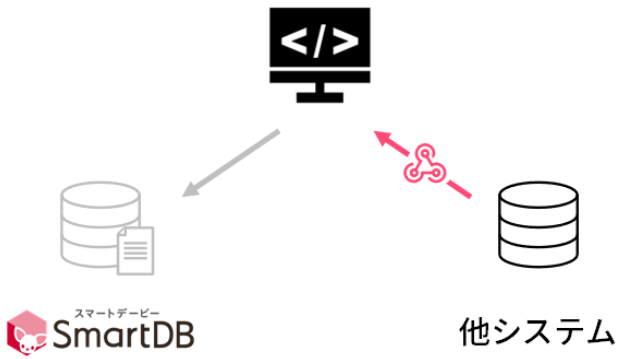
業務プロセスを開始
アクティビティを実施

スマートデービー SmartDB

DreamArts

他システムの処理をトリガーにSmartDBで新規文書のWFを実行

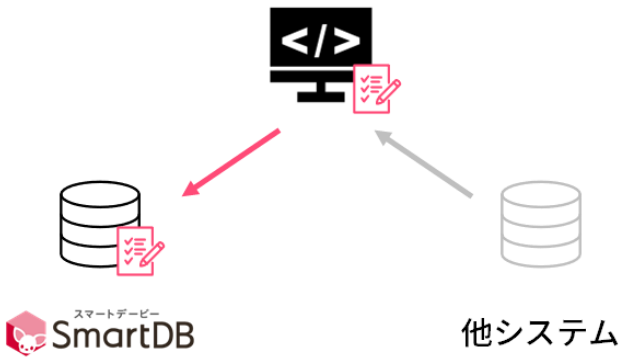
STEP 1



APIの実行プログラムを起動

他システムの提供するWebhook

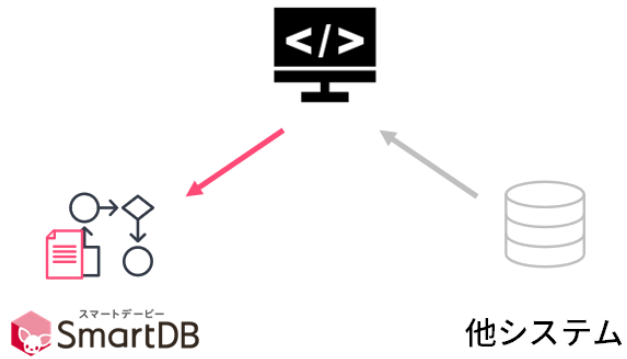
STEP 2



文書の新規登録

文書新規登録

STEP 3



登録した文書のWFを開始

業務プロセス開始

※STEP 2のレスポンスを利用して文書番号を指定

DreamArts

PROCESS

他システムの処理をトリガーに・・・

SmartDBで文書新規登録のWFを実行

他システム

○○という処理を実行

例）人事マスタデータを更新

業務プロセスを開始

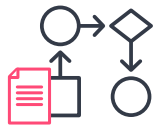
代行設定取得

SmartDBで既存文書のWFを実行

他システム

○○という処理を実行

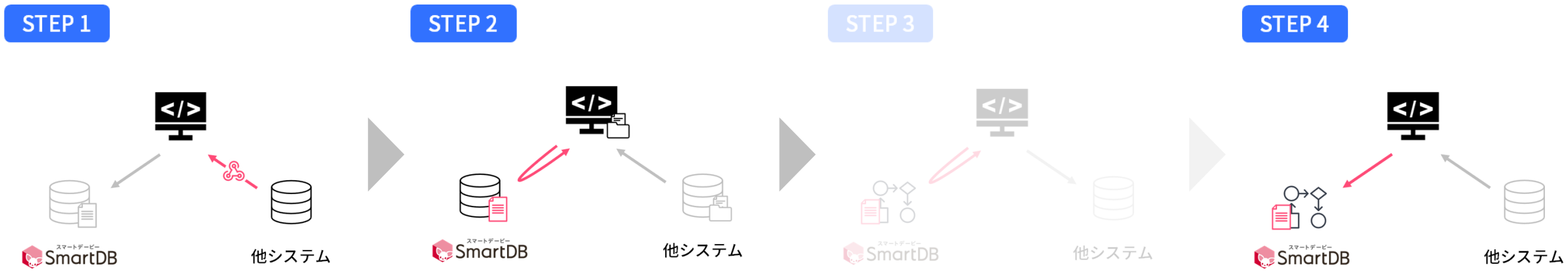
例）人事マスタデータを更新



業務プロセスを開始
アクティビティを実施

DreamArts

他システムの処理をトリガーにSmartDBで既存文書のWFを実行



APIの実行プログラムを起動

他システムのWebhook

WFを開始する文書の文書番号を取得

指定したビューの文書一覧取得

実施するアクティビティのIDを取得

文書情報に基づいて実行中アクティビティ情報取得

既存文書のWFを開始

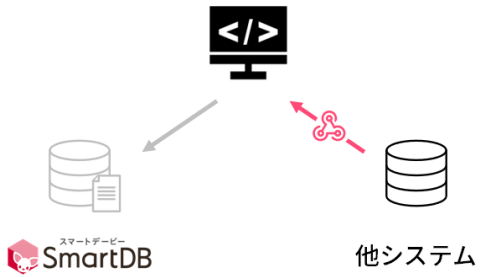
業務プロセス開始

※STEP 4が「業務プロセス開始」の場合は不要

DreamArts

他システムの処理をトリガーにSmartDBで既存文書のWFを実行

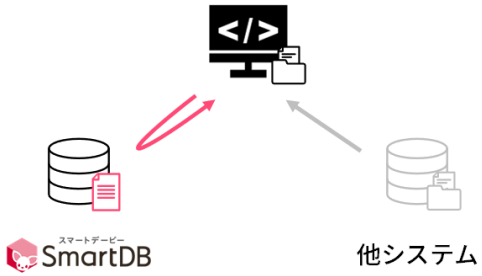
STEP 1



APIの実行プログラムを起動

他システムのWebhook

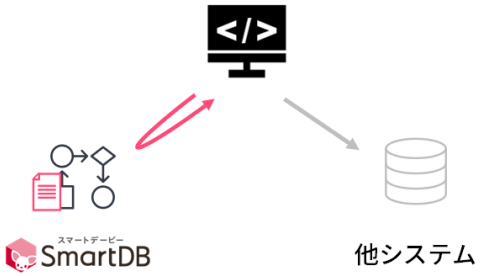
STEP 2



WFを開始する文書の文書番号を取得

指定したビューの文書一覧取得

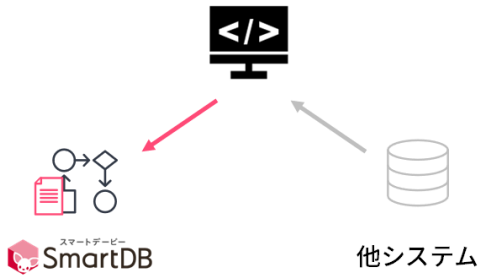
STEP 3



実施するアクティビティのIDを取得

文書情報に基づいて実行中アクティビティ情報取得

STEP 4



既存文書のWFのアクティビティを実施

アクティビティ実施

問題

よく使う業務プロセスのREST APIの説明として正しいのは？

※ DCS / オンプレミスの場合、手順が異なる場合がございます。

1. 業務プロセス開始のREST APIでは、アクティビティIDの指定が必要となる。
2. アクティビティ実施のREST APIでは、アクティビティIDの指定が必要となる。
3. 他システムの処理を起点に、業務プロセスを開始することはできない。
4. 他システムの処理を起点に、アクティビティを実施することはできない。

問題

よく使う業務プロセスのREST APIの説明として正しいのは？

※ DCS / オンプレミスの場合、手順が異なる場合がございます。

1. 業務プロセス開始のREST APIでは、アクティビティIDの指定が必要となる。
2. **アクティビティ実施のREST APIでは、アクティビティIDの指定が必要となる。**
3. 他システムの処理を起点に、業務プロセスを開始することはできない。

解説

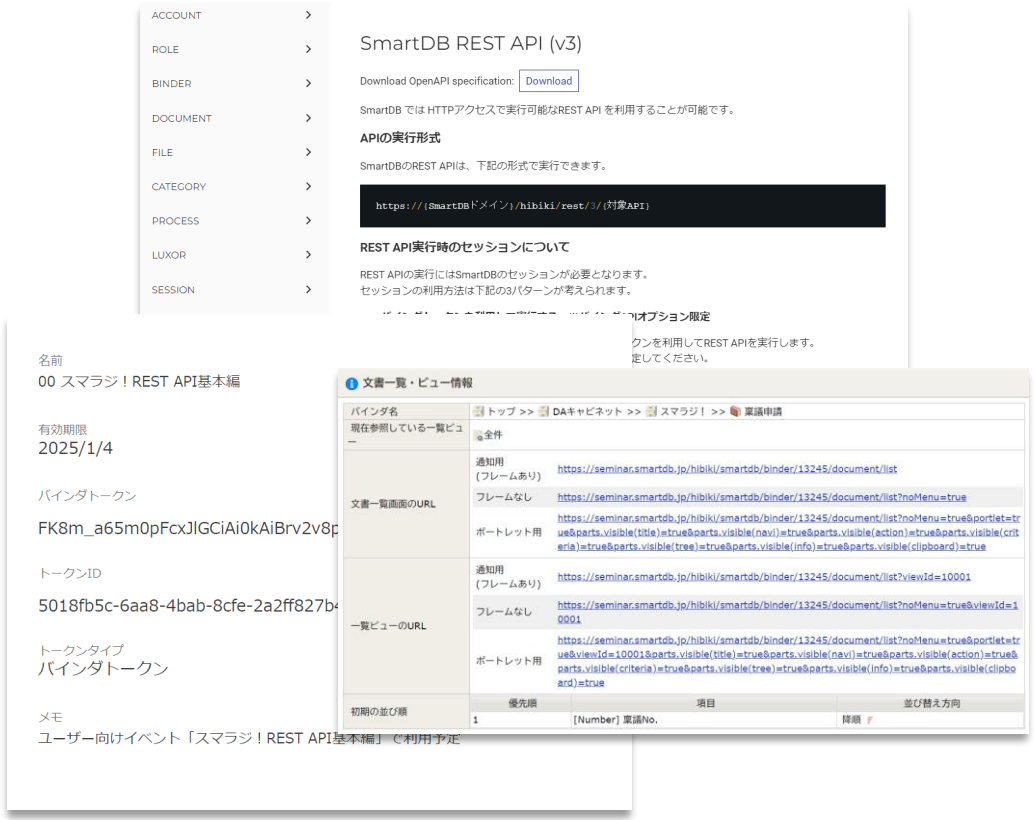
1. 「業務プロセス開始」では、基本的に**プロセス定義番号、開始バインダID（キー）、開始文書の文書ID**によって処理する業務プロセスを指定する。
2. 「アクティビティ実施」では、**アクティビティID**によって処理するアクティビティを指定する。

DreamArts

1. 前回の復習
2. SmartDB REST APIの始め方
3. よく使うSmartDB REST API
4. ハンズオン
5. まとめ
6. 次回予告

DreamArts

参照情報



リクエスト

- URI
- HTTPメソッド
- リクエストヘッダ
- リクエストボディ

[ハンズオンで使用するPowerShellスクリプト]

※コミュニティサイトへの登録が必要です。

参照情報

リクエスト

どのREST APIを使えばいいかわかったけど・・・。

どうやってリクエストを作ればいいの・・・？

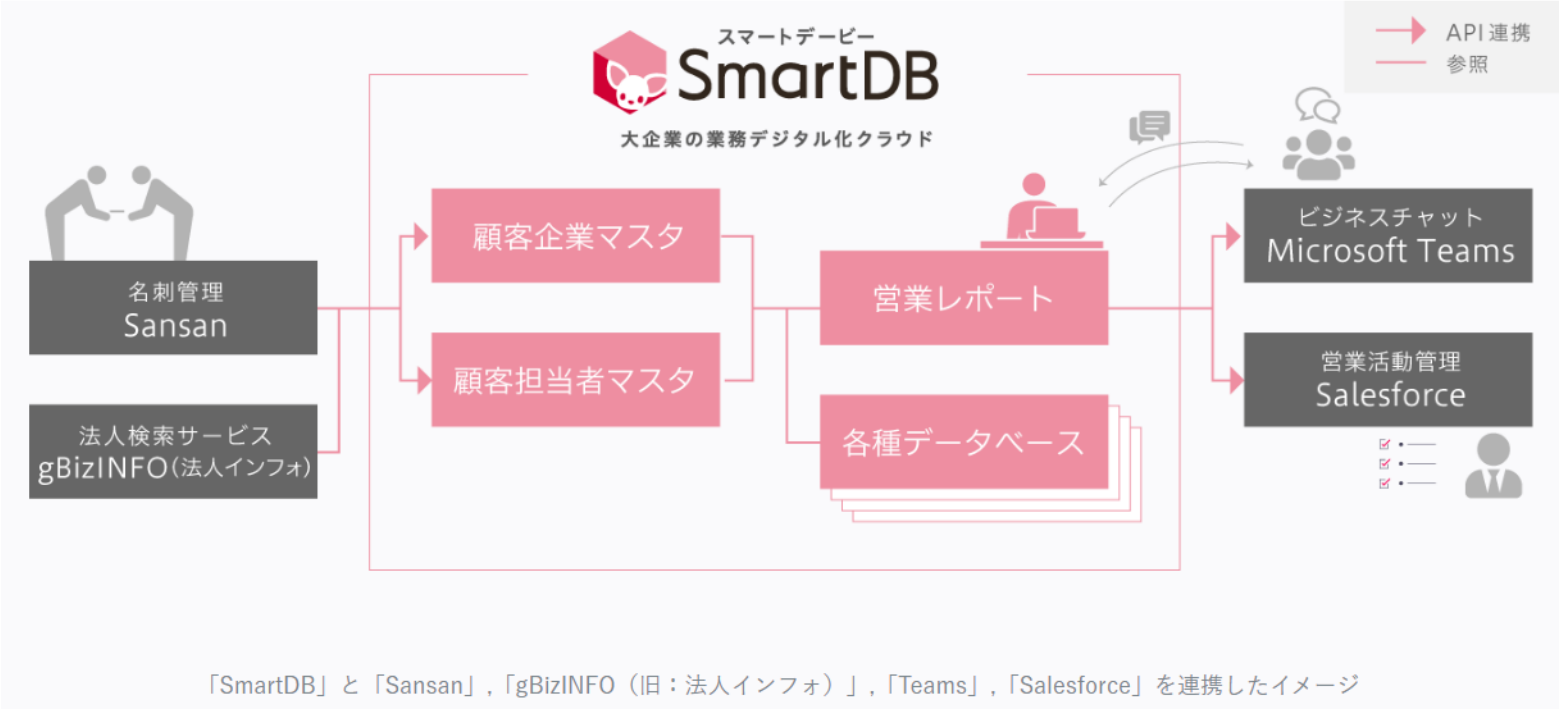
ソッド

リクエストヘッダ

リクエストボディ

DreamArts

他システムの名刺情報からSmartDBの顧客担当者マスタを更新する

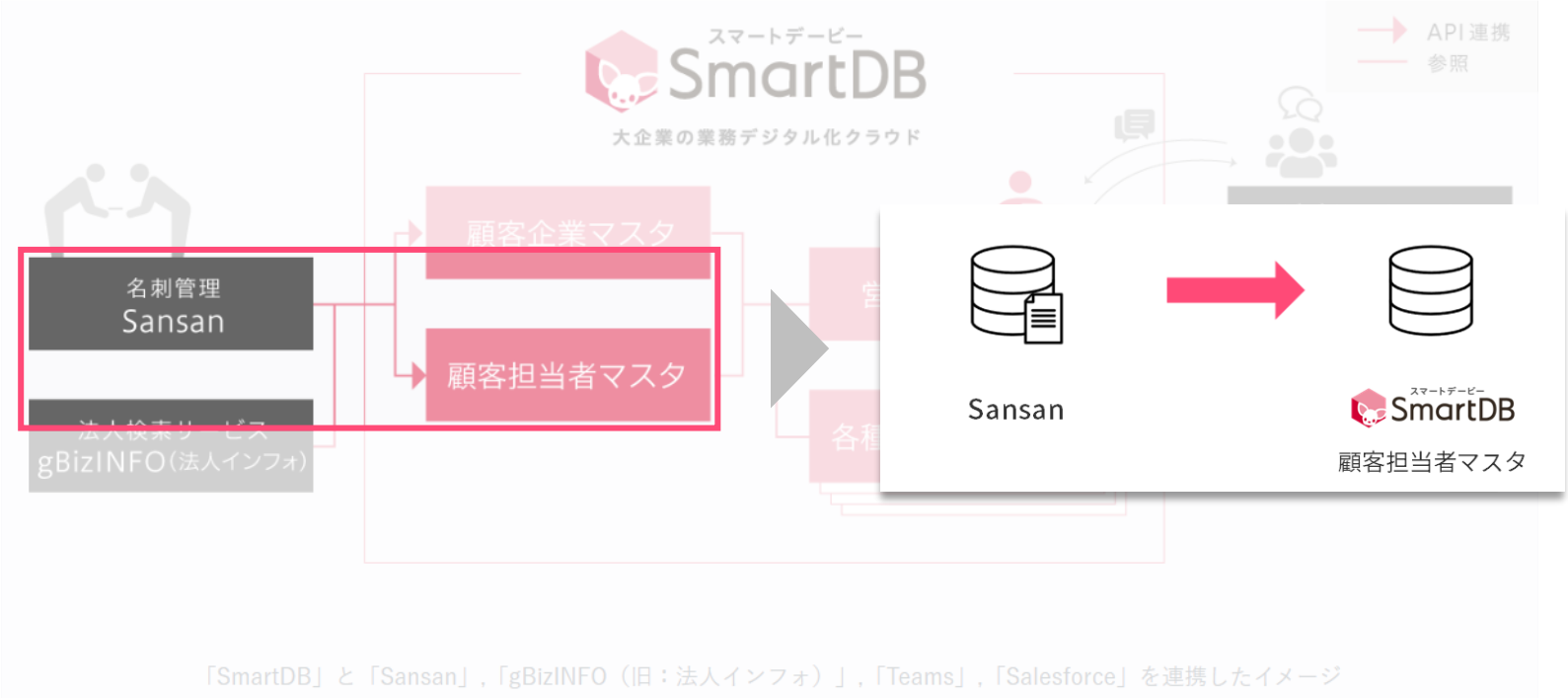


「SmartDB」と「Sansan」,「gBizINFO（旧：法人インフォ）」,「Teams」,「Salesforce」を連携したイメージ

（出典）API連携事例｜「Salesforce」×「Sansan」×「SmartDB」で営業の情報管理効率化｜SmartDB®【大企業の業務デジタル化クラウド】（dreamarts.co.jp）

DreamArts

他システムの名刺情報からSmartDBの顧客担当者マスタを更新する



（出典）API連携事例｜「Salesforce」×「Sansan」×「SmartDB」で営業の情報管理効率化｜SmartDB®【大企業の業務デジタル化クラウド】（dreamarts.co.jp）

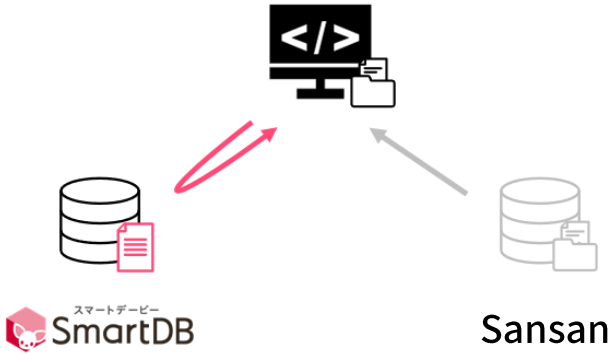
DreamArts

他システムの名刺情報からSmartDBの顧客担当者マスタを更新する

STEP 1

Sansan Open APIを利用して
「人物ID」を取得

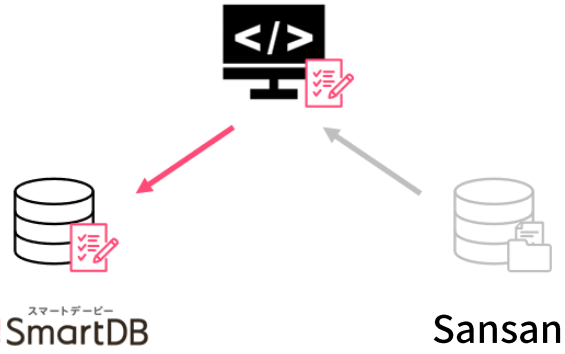
STEP 2



SmartDB REST APIと「人物ID」
を利用して文書番号を取得

※「指定したビューの文書一覧取得」を利用

STEP 3



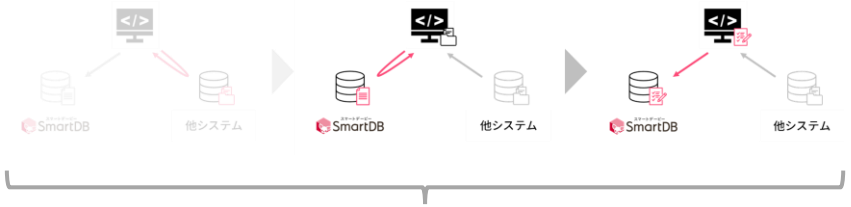
SmartDB REST APIと「文書番号」
を利用して文書のデータを更新

※「文書更新」を利用

【ハンズオン】実行内容の確認（補足）

通常の場合

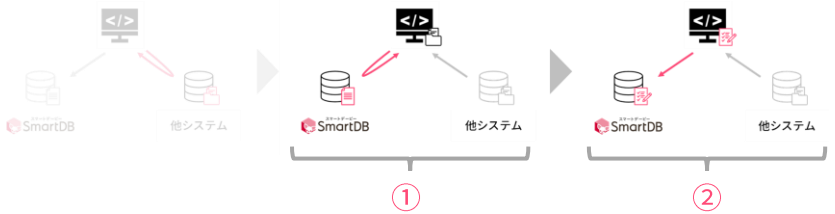
1つのプログラム(コード)で一連の全てSTEPを実行する




```
# 1.サンプルデータ
$SampleData = @{
    "key" = "00080"
    "Title" = "テスト申請（変更）"
}

# 2.文書詳細取得（GET）
$URI = "https://[SmartDB環境ドメイン].smartdb.jp"`
       + "/hibiki/rest/3/binders/[バインダID]/views/[ビューID]/documents"`
       + "?DocNumber=" + $SampleData.key + ":EXACT"
$Method = "GET"
$Headers = @{
            "Authorization" = "Bearer [バインダトークン]"
           }

Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_DocList.json

# 3.2で取得した文書データから文書番号を抽出
$Response = ((Get-Content .¥response_DocList.json -Encoding UTF8) | ConvertFrom-Json)
$DocID = $Response.documents[0].id

# 4.文書更新(PUT)
$URI = "https://[SmartDB環境ドメイン].smartdb.jp"`
       + "/hibiki/rest/3/binders/10608/documents/"`
       + $DocID
$Method = "PUT"
$Headers = @{
            "Authorization" = "Bearer S2Y17UkU0sZOITI7IK6nfH5c0is1N_oh"
           }
$Body = @{
            "DocTitle" = $SampleData.Title
         }
$Body = $Body | ConvertTo-Json
$Body = [System.Text.Encoding]::UTF8.GetBytes($Body)

Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_PUTDoc.json
```

本講座の場合

便宜上、STEPごとにプログラム(コード)を分割する



①

```
$URI = "[①プロトコル＋ホスト名]" `
       + "[②リソースパス]" `
       + "?Title=[③件名]:EXACT"
$Method = "[④HTTPメソッド]"
$Headers = @{
       "[⑤認証用のヘッダ]" = "Bearer [⑥バインダトークン]"
       }
Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_DocList.json
```

＋

②

```
$URI = "[①プロトコル＋ホスト名]" `
       + "[②リソースパス]"
$Method = "[③HTTPメソッド]"
$Headers = @{
       "[④認証用のヘッダ]" = "Bearer [⑤バインダトークン]"
       }
$Body = @{
    "Budget" = [⑥予算額]
    "Reason" = "[⑦起案理由]"
    "Contents" = "[⑧起案内容]"
    }
Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_PUTDoc.json
```

【ハンズオン】実行内容の確認

DreamArts

変更前

変更後

部署異動に伴う名刺の更新情報を顧客担当者マスタに反映

DreamArts

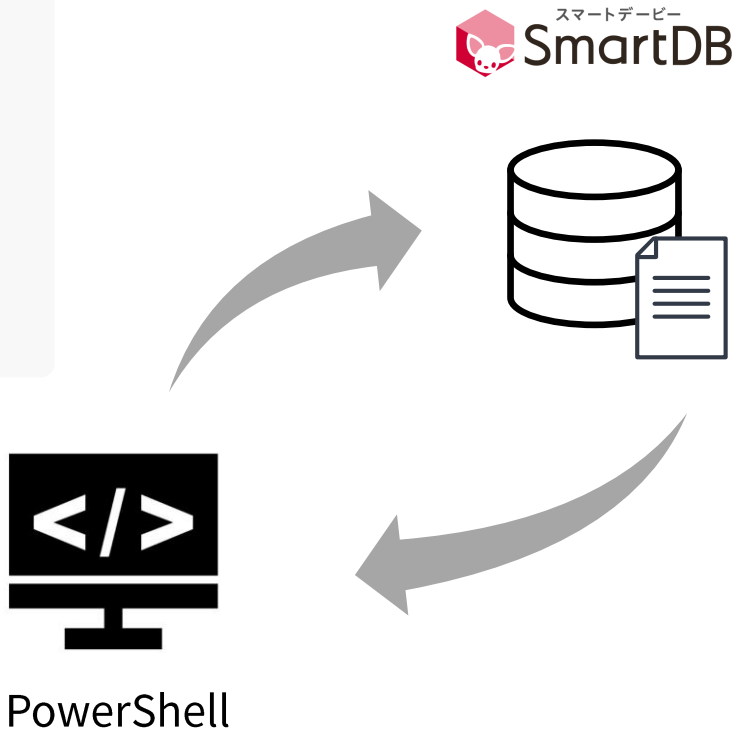
PowerShellでREST APIを実行する方法を確認

1. URI
2. HTTPメソッド
3. リクエストボディ
4. リクエストヘッダ

の指定

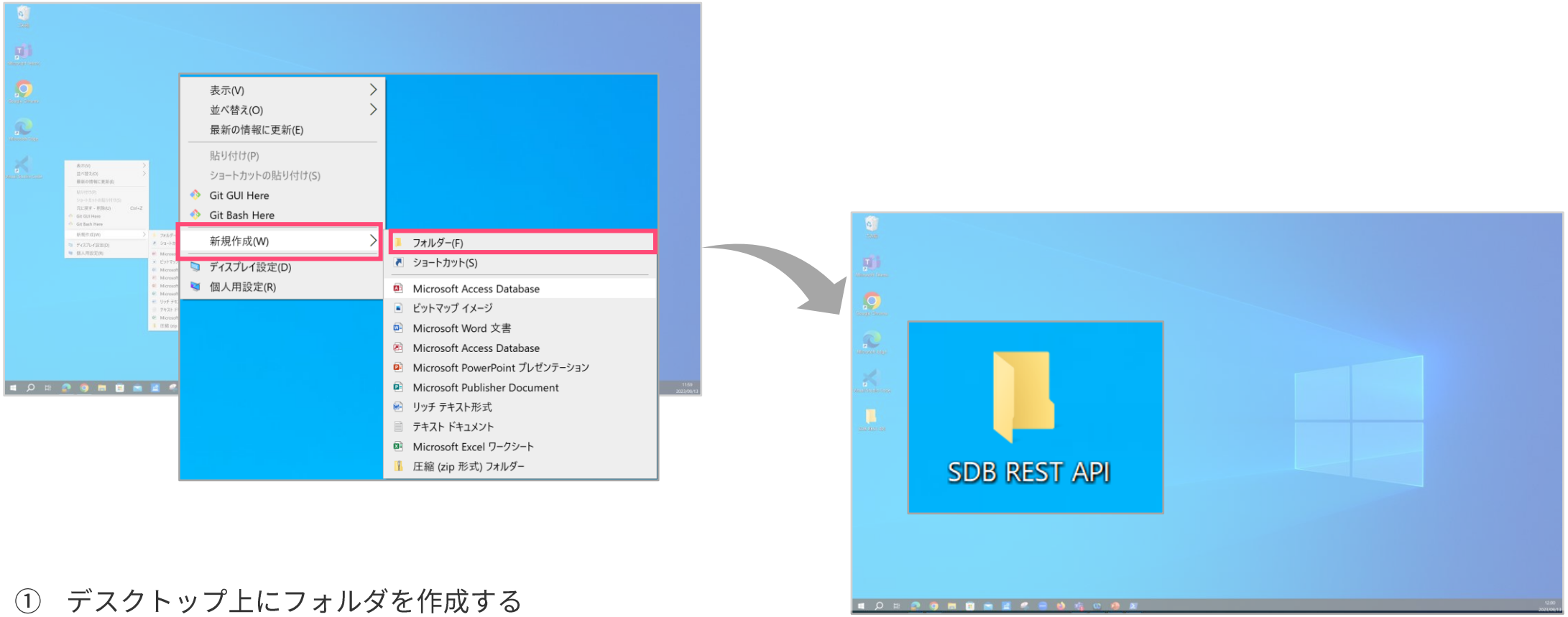
＋

Invoke-WebRequest：REST APIを実行するコマンド



DreamArts

デスクトップにハンズオン用のフォルダを作成



① デスクトップ上にフォルダを作成する
② フォルダに「SDB REST API」と命名する

DreamArts

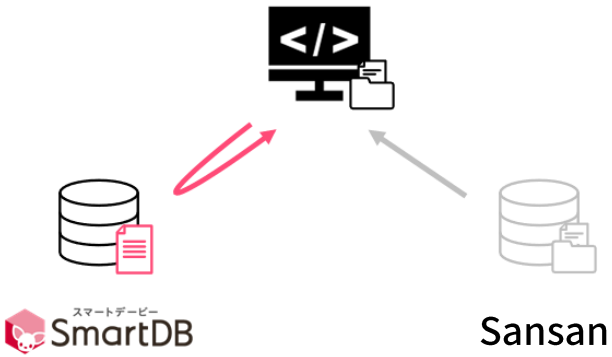
他システムの名刺情報からSmartDBの顧客担当者マスタを更新する

STEP 1

本講座では省略

Sansan Open APIを利用して
「人物ID」を取得

STEP 2



SmartDB REST APIと「人物ID」
を利用して文書番号を取得

※「指定したビューの文書一覧取得」を利用

STEP 3

SmartDB REST APIと「文書番号」
を利用して文書のデータを更新

※「文書更新」を利用

DreamArts

URIのプロトコル、リソースパスを指定

RestDocV3

リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]” `
        + “?personalId=[②人物ID]:EXACT”
$Method = “[③HTTPメソッド]"
$Headers = @{
        “[④認証用のヘッダ]” = “
        "Content-type"="applic
        }
Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_DocList.json
```

```
$URI = “[①プロトコル + ホスト名+リソースパス]” `
        + “?personalId=[②人物ID]:EXACT”
```

```
$URI = “https://{hostname}.smartdb.jp/hibiki/rest/3/binders/{binder}/views/{view}/documents” `
        + “?personalId=[②人物ID]:EXACT”
```

DreamArts

URIのホスト名、リソースパラメータを指定

SmartDBの文書一覧情報

リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]” `
        + “?personalId=[②人物ID]:EXACT”
$Method = “[③HTTPメソッド]"
```

```
$URI = “https://{hostname}.smartdb.jp/hibiki/rest/3/binders/{binder}/views/{view}/documents” `
        + “?personalId=[②人物ID]:EXACT”
```

```
Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_DocList.json
```

```
$URI = “https://seminar.smartdb.jp/hibiki/rest/3/binders/13243/views/10004/documents” `
        + “?personalId=[②人物ID]:EXACT”
```

DreamArts

URIのクエリパラメータを指定

他システムから取得した元データ

変更後

項目	値
人物ID	AA0001
氏名	夢野 芸
会社名	株式会社ドリーム・アーツ
部署名	経営企画部
	⋮

リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]” `
        + “?personalId=[②人物ID]:EXACT”
$Method = “[③HTTPメソッド]"
$Headers = @{
```

```
$URI = “https://seminar.smartdb.jp/hibiki/rest/3/binders/13243/views/10004/documents” `
        + “?personalId=[②人物ID]:EXACT”
```

```
Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_DocList.json
```

```
$URI = “https://seminar.smartdb.jp/hibiki/rest/3/binders/13243/views/10004/document” `
        + “?personalId=AA0001:EXACT”
```

DreamArts

HTTPメソッドを指定

RestDocV3

リクエスト

```
$URI = “[①プロトコル＋ホスト名+リソースパス]” `
        + “?personalId=[②人物ID]:EXACT”
$Method = “[③HTTPメソッド]"
$Headers = @{
        “[④認証用のヘッダ]” = “Bearer [⑤バインダトークン]“
        "Content-type"="application/jso
        }
Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_DocList.json
```

$Method = “[③HTTPメソッド]"

$Method = “GET"

DreamArts

HTTPヘッダの認証情報の方式を指定

RestDocV3

REST API実行時のセッションについて

REST APIの実行にはSmartDBのセッションが必要となります。
セッションの利用方法は下記の3パターンが考えられます。

- **バインダトークンを利用して実行する。※バインダAPIオプション限定**

  管理画面の「バインダトークン」画面で発行したトークンを利用してREST APIを実行します。
  実行時には後述の形式でHTTP Headerにトークンを指定してください。
  バインダトークンを利用してAPIを実行する場合、下記の特徴を持ちます。

  - トークンを実行するためのロボットユーザとして実行されます。
    ※ロボットユーザのタイムゾーンは+0900
  - ParameterのcsrfTokenは不要です。
  - バインダを指定するAPIの場合、トークンを設定したバインダにのみ権限

```
Authorization: Bearer {トークン}
```

HTTPヘッダーに以下を指定してください。

```
Authorization: Bearer {トークン}
```

リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]” `
    + “?personalId=[②人物ID]:EXACT”
$Method = “[③HTTPメソッド]"
$Headers = @{
    “[④認証用のヘッダ]” = “Bearer [⑤バインダトークン]“
    "Content-type"="application/json"
    }
Invoke-WebRequest $URI -Meth                                    ile .¥response_DocList.json
```

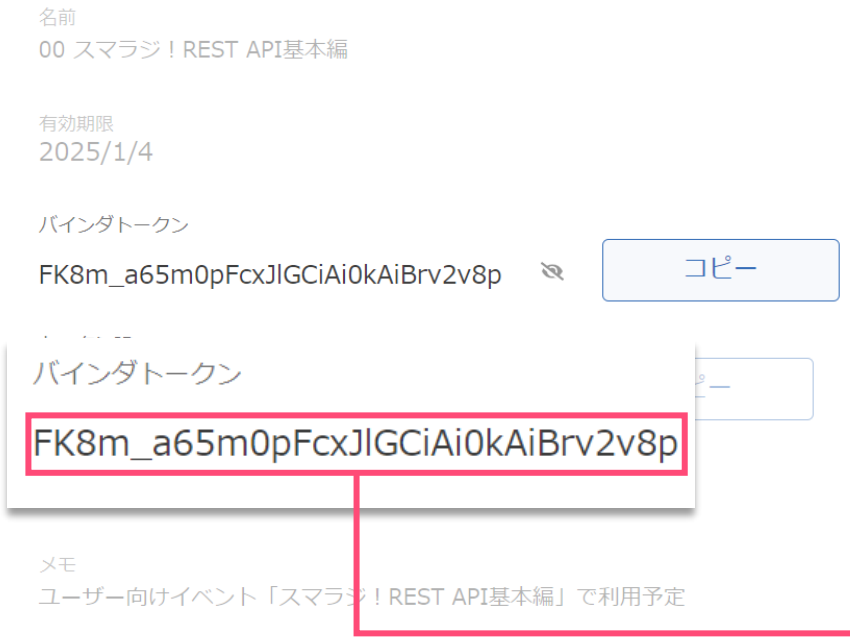
```
$Headers = @{
    “[④認証用のヘッダ]” = “Bearer [⑤バインダトークン]“
    "Content-type"="application/json"
    }
```

```
$Headers = @{
    “Authorization” = “Bearer [⑤バインダトークン]“
    "Content-type"="application/json"
    }
```

DreamArts

認証情報のバインダトークンを指定

SmartDBのバインダトークン管理画面



リクエスト

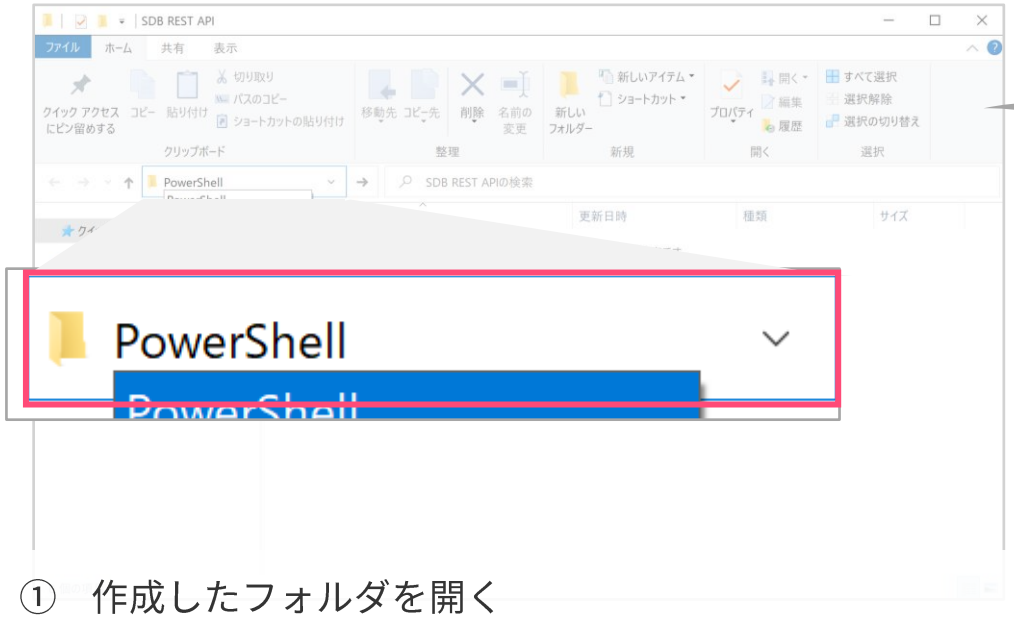
```
$URI = “[①プロトコル＋ホスト名+リソースパス]” `
        + “?personalId=[②人物ID]:EXACT”
$Method = “[③HTTPメソッド]"
$Headers = @{
      “[④認証用のヘッダ]” = “Bearer [⑤バインダトークン]“
      "Content-type"="application/json"
      }
Invoke-WebRequest $URI -Meth                                    File .¥response_DocList.json
```

```
$Headers = @{
      “Authorization” = “Bearer [⑤バインダトークン]“
      "Content-type"="application/json"
      }
```

```
$Headers = @{
      “Authorization” = “Bearer FK8m_a65m0pFcxJlGCiAi0kAiBrv2v8p“
      "Content-type"="application/json"
      }
```

DreamArts

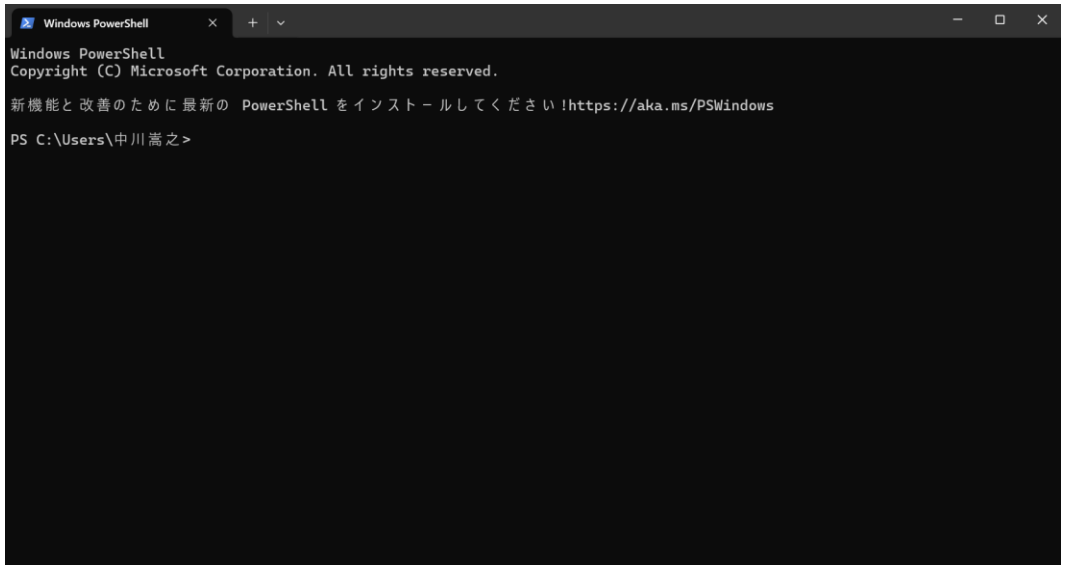
作成したフォルダからPowerShellを起動



① 作成したフォルダを開く

② アドレスバーに「PowerShell」と入力する

③ Enterキーを押す



※ログファイルの出力先を今回作成したフォルダにするため、フォルダから起動しています。

DreamArts

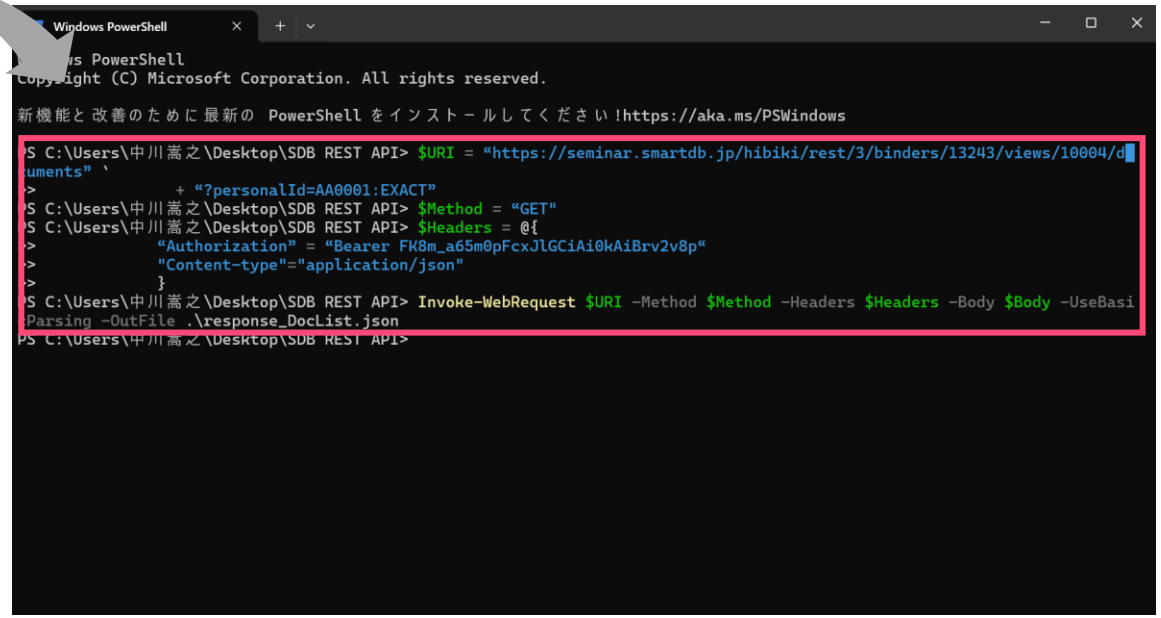
作成したコードをPowerShellで実行

【記入用】文書番号を取得するコード

DreamArts

```
$URI = “https://seminar.smartdb.jp/hibiki/rest/3/binders/13243/views/10004/documents” `
        + “?personalId=AA0001:EXACT”
$Method = “GET"
$Headers = @{
        “Authorization” = “Bearer FK8m_a65m0pFcxJlGCiAi0kAiBrv2v8p“
        "Content-type"="application/json"
        }
Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_DocList.json
```

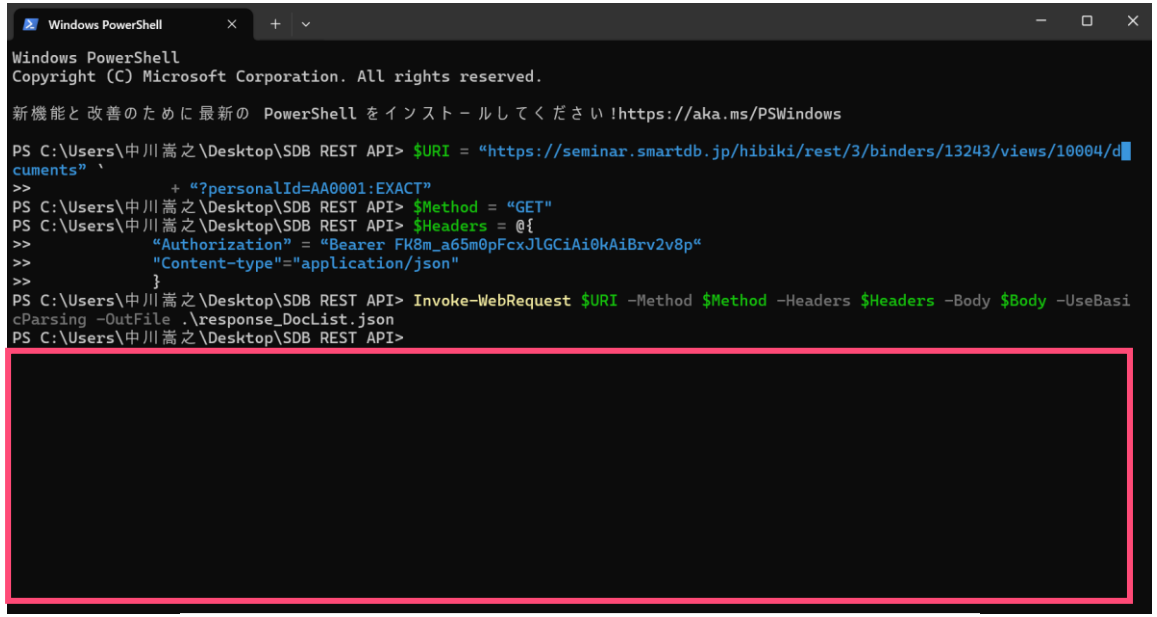

① 作成したコードをコピーする
② PowerShellに貼り付ける
③ Enterキーで実行する



DreamArts

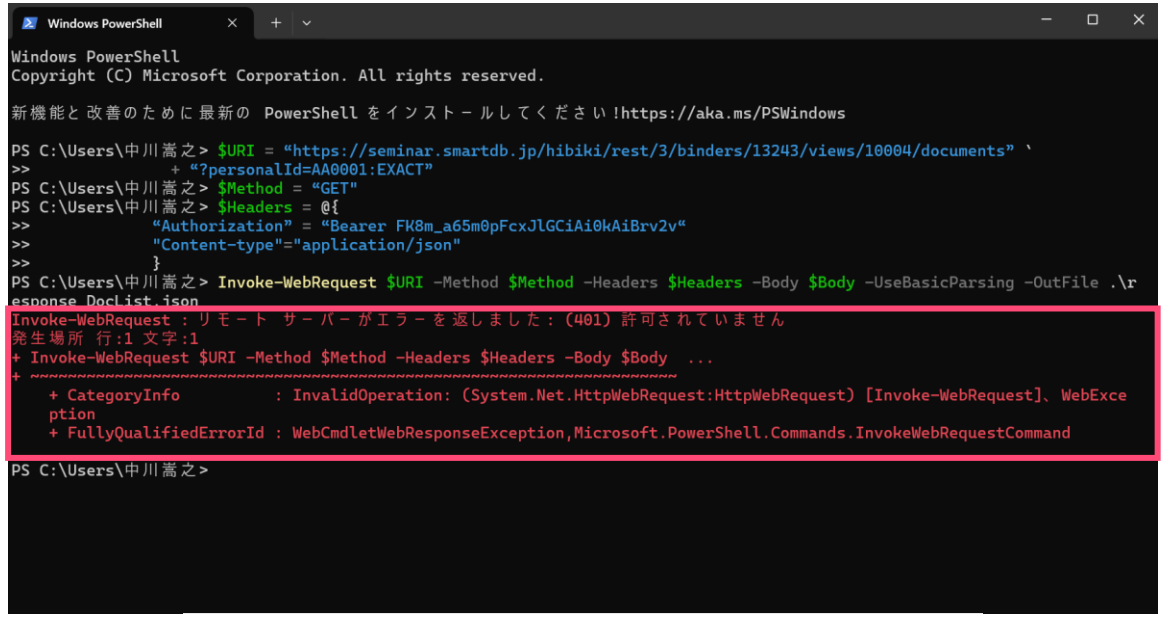
実行結果を確認

成功



エラーが出力されない

失敗



赤字でエラーが出力

DreamArts

他システムの名刺情報からSmartDBの顧客担当者マスタを更新する

STEP 1

本講座では省略

Sansan Open APIを利用して
「人物ID」を取得

STEP 2

SmartDB

Sansan

SmartDB REST APIと「人物ID」
を利用して文書番号を取得

※「指定したビューの文書一覧取得」を利用

STEP 3

SmartDB

Sansan

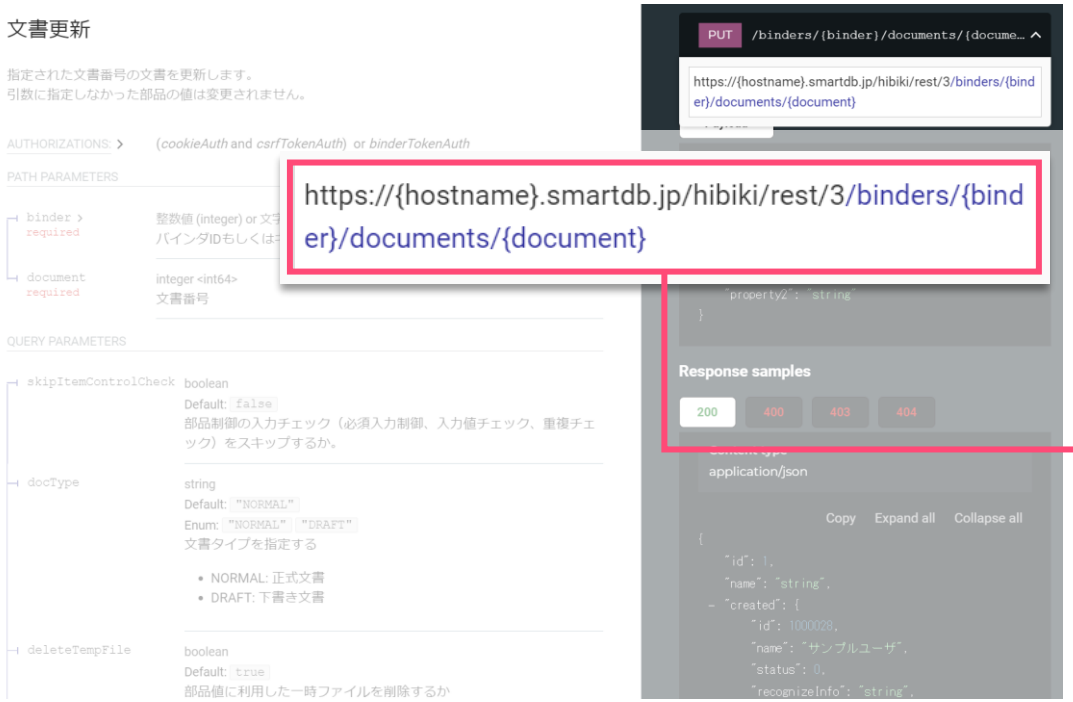
SmartDB REST APIと「文書番号」
を利用して文書のデータを更新

※「文書更新」を利用

DreamArts

URIのプロトコル、リソースパスを指定

RestDocV3



リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]”
$Method = “[②HTTPメソッド]"
$Headers = @{
    “[③認証用のヘッダ]” = “Bearer [④バインダトークン] “
    "Content-type"="app
    }
$Body = @{
    “[⑤部署名の部品キー]” = “[⑥部署名の値]"
    }
$Body = $Body | ConvertTo-Json
```

$URI = “[①プロトコル + ホスト名+リソースパス]”

$URI = “https://{hostname}.smartdb.jp/hibiki/rest/3/binders/{binder}/documents/{document}”

DreamArts

URIのホスト名、リソースパラメータを指定

SmartDBの文書一覧情報

リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]”
$Method = “[②HTTPメソッド]"
$Headers = @{
    “[③認証用のヘッダ]” = “Bearer [④バインダトークン] “
}
$URI = “https://{hostname}.smartdb.jp/hibiki/rest/3/binders/{binder}/documents/{document}”
$Body = @{
    “[⑤部署名の部品キー]” = “[⑥部署名の値]"
    }
$Body = $Body | ConvertTo-Json
$URI = “https://seminar.smartdb.jp/hibiki/rest/3/binders/13243/documents/{document}”
```

DreamArts

URIのリソースパラメータを指定

前のSTEPで取得したデータ

リクエスト

$URI = “[①プロトコル + ホスト名+リソースパス]”
$Method = “[②HTTPメソッド]"
$Headers = @{
“[③認証用のヘッダ]” = “Bearer [④バインダトークン] “

$URI = “https://seminar.smartdb.jp/hibiki/rest/3/binders/13243/documents/{document}”

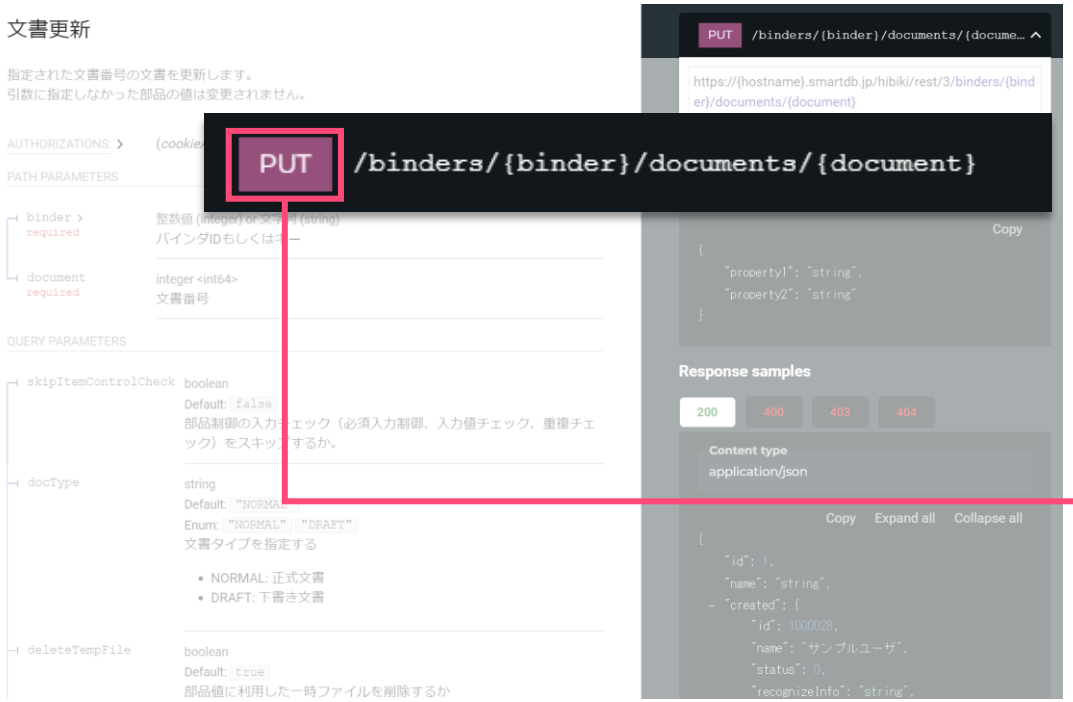
$Body = @{
“[⑤部署名の部品キー]” = “[⑥部署名の値]"
}
$Body = $Body | ConvertTo-Json

$URI = “https://seminar.smartdb.jp/hibiki/rest/3/binders/13243/documents/1”

DreamArts

HTTPメソッドを指定

RestDocV3



リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]”
$Method = “[②HTTPメソッド]"
$Headers = @{
    “[③認証用のヘッダ]” = “Bearer [④バインダトークン] “
    "Content-type"="application/
    }
$Body = @{
    “[⑤部署名の部品キー]” = “[⑥部署名の値]"
    }
$Body = $Body | ConvertTo-Json
$Body = [System.Text.Encoding]::UTF8.GetBytes
Invoke-WebRequest $URI -Method $Method -He                    BasicParsing -OutFile .¥response_PUTDoc.json
```

$Method = “[②HTTPメソッド]"

$Method = “PUT"

DreamArts

HTTPヘッダの認証情報の方式を指定

RestDocV3

REST API実行時のセッションについて

REST APIの実行にはSmartDBのセッションが必要となります。
セッションの利用方法は下記の3パターンが考えられます。

- **バインダトークンを利用して実行する。※バインダAPIオプション限定**

  管理画面の「バインダトークン」画面で発行したトークンを利用してREST APIを実行します。
  実行時には後述の形式でHTTP Headerにトークンを指定してください。
  バインダトークンを利用してAPIを実行する場合、下記の特徴を持ちます。

  - トークンを実行するためのロボットユーザとして実行されます。
    ※ロボットユーザのタイムゾーンは+0900
  - Parameterのcsrfは不要です。
  - バインダを指定するAPIの場合、トークンを設定したバインダにのみ権限

```
Authorization: Bearer {トークン}
```

HTTPヘッダーに以下を指定してください。

```
Authorization: Bearer {トークン}
```

リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]”
$Method = “[②HTTPメソッド]"
$Headers = @{
    “[③認証用のヘッダ]” = “Bearer [④バインダトークン] “
    "Content-type"="application/json"
    }
$Body = @{
    “[⑤部署名の部品キー]” =
    }
$Body = $Body | ConvertTo-Jso
$Body = [System.Text.Encoding
Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_PUTDoc.json
```

```
$Headers = @{
    “[④認証用のヘッダ]” = “Bearer [⑤バインダトークン]“
    "Content-type"="application/json"
    }
```

```
$Headers = @{
    “Authorization” = “Bearer ⑤バインダトークン]“
    "Content-type"="application/json"
    }
```

DreamArts

認証情報のバインダトークンを指定

SmartDBのバインダトークン管理画面

リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]”
$Method = “[②HTTPメソッド]"
$Headers = @{
      “[③認証用のヘッダ]” = “Bearer [④バインダトークン] “
      "Content-type"="application/json"
      }
$Body = @{
      “[⑤部署名の部品キー]” =
      }
$Body = $Body | ConvertTo-Jso
$Body = [System.Text.Encoding
Invoke-WebRequest $URI -Method $Method -Headers $Headers -Body $Body -UseBasicParsing -OutFile .¥response_PUTDoc.json
```

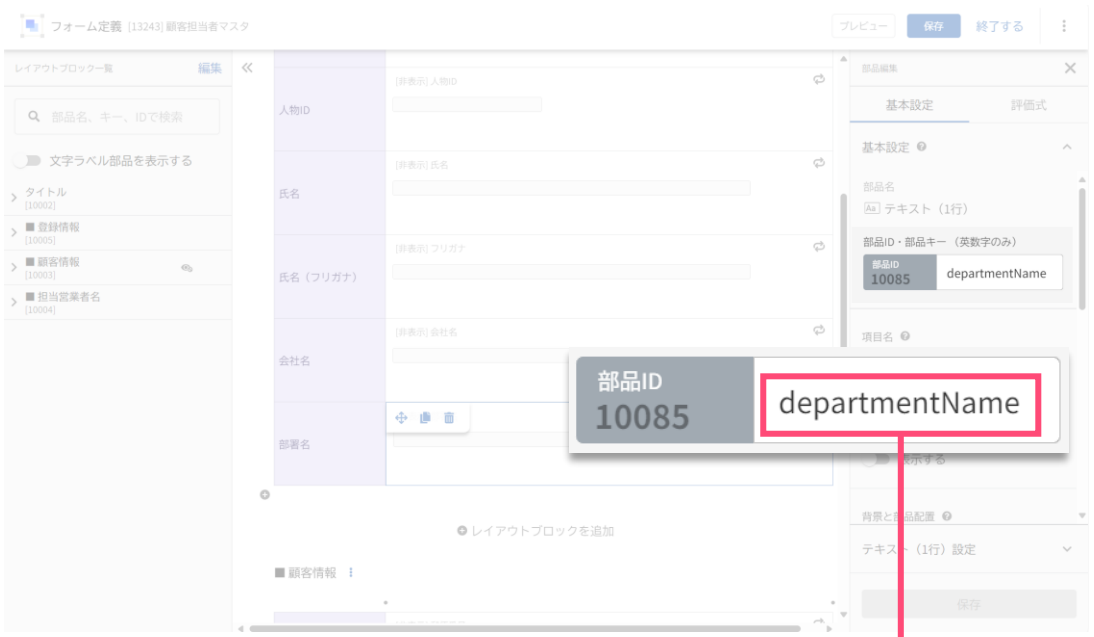
```
$Headers = @{
      “Authorization” = “Bearer [⑤バインダトークン]“
      "Content-type"="application/json"
      }
```

```
$Headers = @{
      “Authorization” = “Bearer FK8m_a65m0pFcxJlGCiAi0kAiBrv2v8p “
      "Content-type"="application/json"
      }
```

DreamArts

リクエストボディの項目を指定

SmartDBの新フォーム定義



リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]”
$Method = “[②HTTPメソッド]"
$Headers = @{
      “[③認証用のヘッダ]” = “Bearer [④バインダトークン] “
      "Content-type"="application/json"
      }
$Body = @{
      “[⑤部署名の部品キー]” = “[⑥部署名の値]"
      }
$Body = $Body | ConvertTo-Json
$Body = [System.Text.Encoding]::UTF8.GetByt
Invoke-WebRequest $URI -Method $Method -H
```

```
$Body = @{
      “[⑤部署名の部品キー]” = “[⑥部署名の値]"
      }
```

```
$Body = @{
      “departmentName” = “[⑥部署名の値]"
      }
```

DreamArts

リクエストボディの項目を指定

取得した文書データ

変更後

項目	値
人物ID	AA0001
氏名	夢野 芸
会社名	株式会社ドリーム・アーツ
部署名	経営企画部
	⋮

リクエスト

```
$URI = “[①プロトコル + ホスト名+リソースパス]”
$Method = “[②HTTPメソッド]"
$Headers = @{
    “[③認証用のヘッダ]” = “Bearer [④バインダトークン] “
    "Content-type"="application/json"
    }
$Body = @{
    “[⑤部署名の部品キー]” = “[⑥部署名の値]"
    }
$Body = $Body | ConvertTo-Json
$Body = [System.Text.Encoding]::UTF8.GetByt
Invoke-WebRequest $URI -Method $Method -H
```

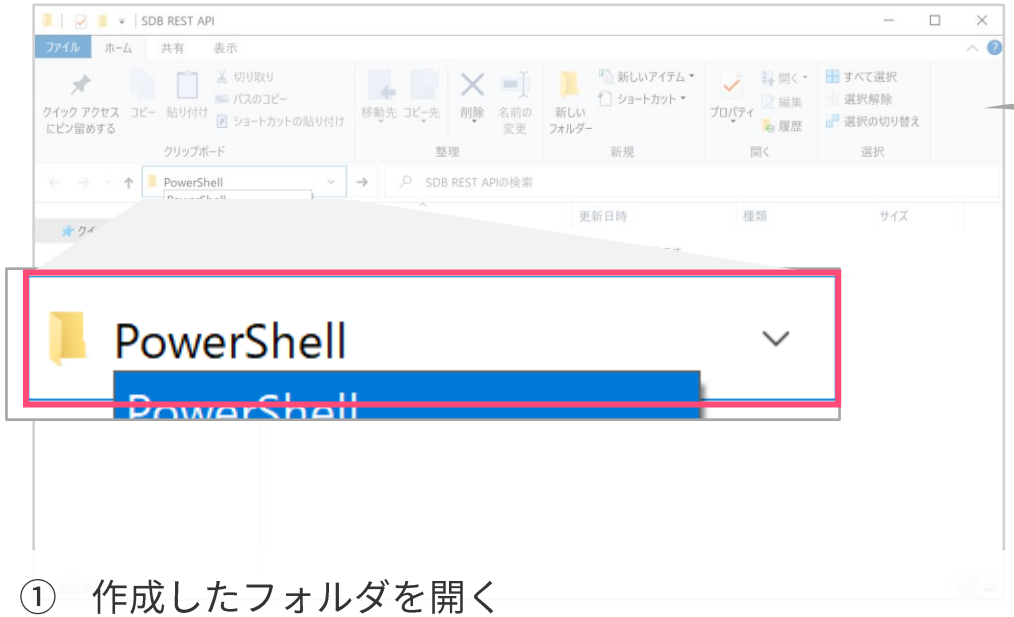
```
$Body = @{
    “departmentName” = “[⑥部署名の値]"
    }
```

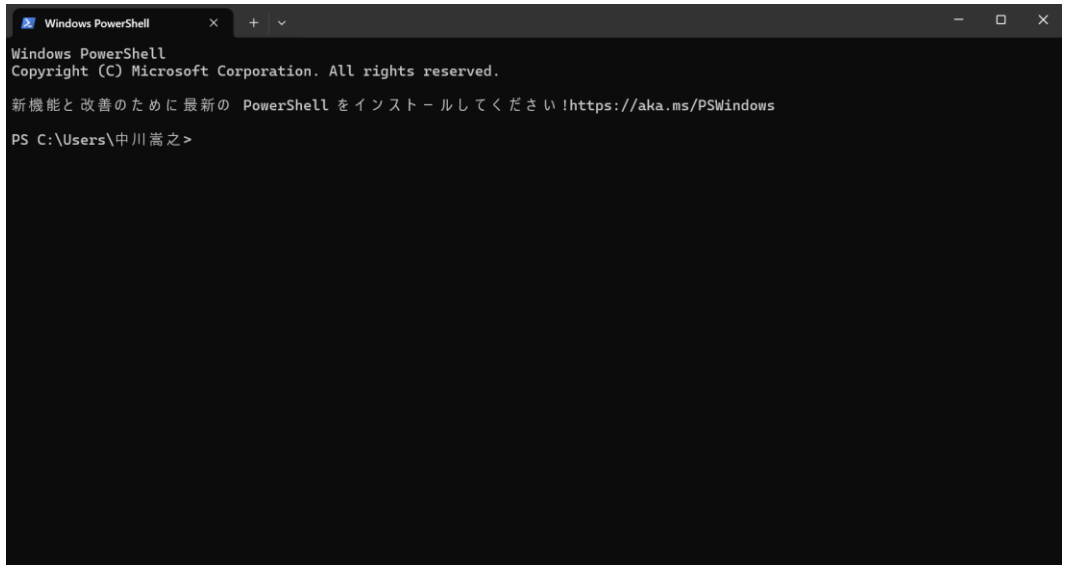
_PUTDoc.json

```
$Body = @{
    “departmentName” = “経営企画部"
    }
```

DreamArts

作成したフォルダからPowerShellを起動





① 作成したフォルダを開く

② アドレスバーに「PowerShell」と入力する

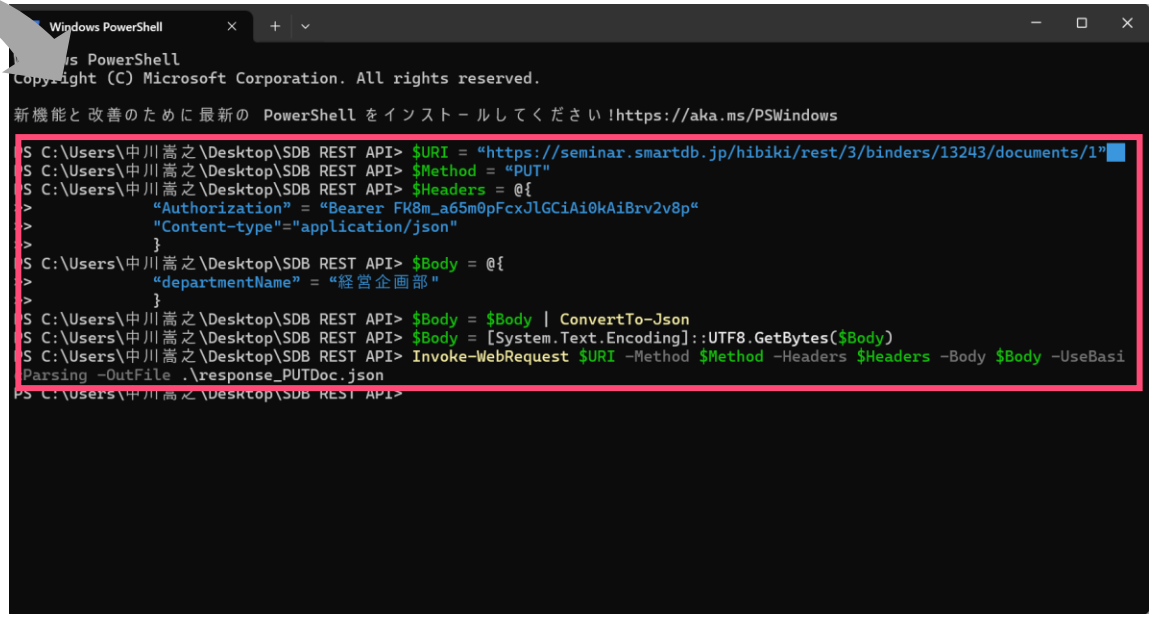
③ Enterキーを押す

※ログファイルの出力先を今回作成したフォルダにするため、フォルダから起動しています。

DreamArts

作成したコードをPowerShellで実行

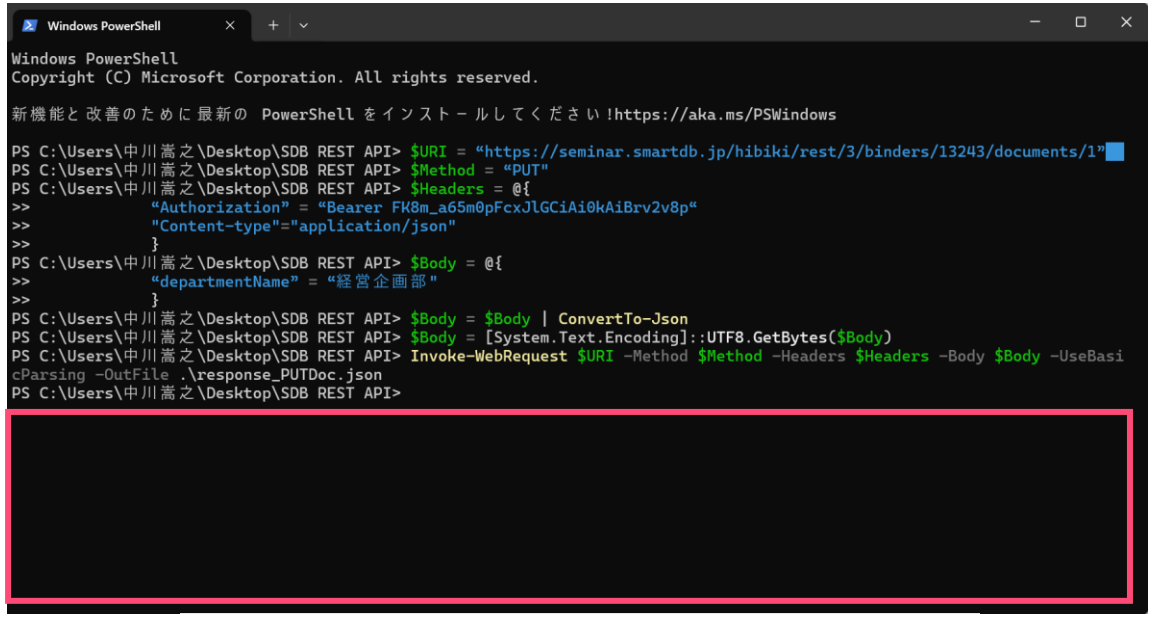
① 作成したコードをコピーする
② PowerShellに貼り付ける
③ Enterキーで実行する



DreamArts

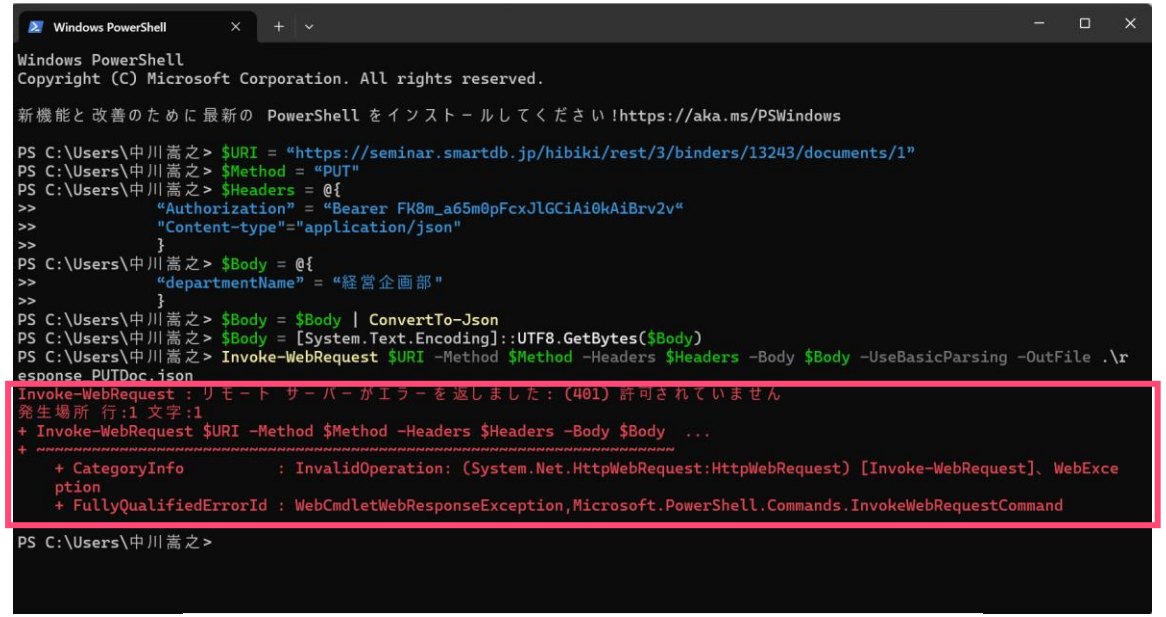
実行結果を確認

成功



エラーが出力されない

失敗



赤字でエラーが出力

【ハンズオン】実行結果の確認

DreamArts

変更前

変更後

DreamArts

1. 前回の復習
2. SmartDB REST APIの始め方
3. よく使うSmartDB REST API
4. ハンズオン
5. まとめ
6. 次回予告

1. SmartDB REST APIの利用には、バインダトークンが必要
2. RestDocV3、SmartDB環境情報を参照してリクエスト作成
3. SmartDB REST APIを組み合わせることで業務を実現
4. 最もよく使うSmartDB REST APIは下表の通り

カテゴリ	REST API
バインダ / 文書	文書詳細情報取得 指定したビューの文書一覧取得 文書一覧のCSV出力 文書新規登録 文書更新 文書削除 文書一括操作 CSV入力（新規） CSV入力（更新）
業務プロセス	業務プロセス開始 アクティビティ実施 文書情報に基づいて実行中アクティビティ情報取得

DreamArts

1. 前回の復習
2. SmartDB REST APIの始め方
3. よく使うSmartDB REST API
4. ハンズオン
5. まとめ
6. 次回予告

次回予告

No.	実施概要	詳細内容	推奨者
1	REST API 入門編	REST API概説 実践：PowerShellでREST APIを実行してみよう	SmartDB利用経験のある方
本日	SmartDB REST API アプリ操作基本編	SmartDB REST APIの基本的な使い方 よく使う基本的なAPIの解説と実践	1の参加者、もしくは1の内容を理解している方
3/26	SmartDB REST API アプリ操作応用編	SmartDB REST APIの応用的な使い方 複数のREST APIを用いた業務適用の解説と実践	2の参加者、もしくは2の内容を理解している方
4	SmartDB REST API アカウンタマスタ連携編	アカウントAPIの使い方 アカウントAPIを用いた業務適用の解説と実践	1の参加者、もしくは1の内容を理解している方
5	SmartDB REST API 業務・性能・セキュリティ編	業務影響を軽減するための設計 SDBの性能を考慮した設計・運用	3の参加者、もしくは3の内容を理解している方 または、 4の参加者、もしくは4の内容を理解している方
6	SmartDB REST API システム連携設計編	SmartDBを含む複数システム連携の全体設計 SDB起点の実行方法：Webhook、定期バッチ処理	5の参加者、もしくは5の内容を理解している方

※上記の内容は変更される場合がございます

お知らせ

資料や動画は、後日コミュニティサイト「スマラジ！ルーム」へ掲載します。

本イベントに関するご質問も受付けますのでお気軽に投稿ください。

https://cs.support-smartdb.com/hc/ja/community/topics/5423952120601

※閲覧にはサポートサイトへのログインが必要です。

DreamArts

コミュニティサイトでは、DAや他ユーザーさんへの質問が可能です。
気になったことがあれば、お気軽に投稿ください。

スマートデービー SmartDB サポートサイト

サイト内

SmartDBサポートサイト ＞ コミュニティ ＞ 雑談・つぶやき

非推奨のログイン方法だけど・・・SmartDB REST APIをExcel VBAで試した話

あおさん（趣味C#プログラマー） Great supporter First post

2023年09

前回に続いて、API関連の話を書いてみます。
但し、今回のログイン方法は**公式に非推奨**であり、
セキュリティ的によろしくなく、
またいつ使えなくなってもおかしくないのでご注意ください。
また、複雑になるのでVBAのコードは書きません。

まず、最新のAPI（V3）を確認してみます。

SmartDB REST API (support-dreamarts.com)

セッションについての説明の3つ目を見ます

- `post /session` を実行して、セッションを作成する。※非推奨

  ID/PASSWORDを指定してsession APIを実行してセッションを作成します。
  ここで作成したセッションを利用して、その他のREST APIを実行します。バッチ処理などを特定ユーザで実行したい場合などです。
  プログラム中にユーザのID/パスワードを記述する必要があるため、セキュリティの観点からも非推奨とします。

正直この説明が全てです。
バインダAPIオプション契約前のテストぐらいには使えるかもしれませんが、
セキュリティ的に本運用には使えません。

まず、この方法がまだ使えるのか確認します。
お手軽にWindowsのコマンドプロンプトからcurlでやってみます。
ログインが、

SmartDBサポートサイト ＞ コミュニティ ＞ 一般Q&A

グラフダイアログのカラー設定

フォローする

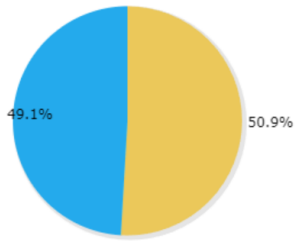
のほほん Great supporter Nice multiple posts

202

皆さん、グラフダイアログって利用されていますか？
簡易なグラフなら便利かな？と思って設定していたのですが
値の大きい順に色設定されてしまう事に気が付きました。
これって、あたり前の事なのでしょうか(゜Д゜)

車の運転前と運転後にそれぞれ登録するバインダがあります。
運転前の登録数と運転後の登録数は原則イコールになる想定なのですが
明らかに登録割合のおかしい人には注意をしようと思ってグラフを表示しました。

●Aさん



運転前後	部品件数	比率
運転後	55	50.9%
運転前	53	49.1%
合計	108	100.0%

運転前と運転後がほぼ同じなので、よし

SmartDBサポートサイト ＞ コミュニティ ＞ What's New！

スマコミュ！マガジン_vol.31～10/18デジタルの民主化DAYを開催しました！～

フォローする

廣瀬 璃奈 DreamArts

2023年11月09日 14:14

0

■◆■――――――――――――――― vol.31―2023.11.09―

スマコミュ！マガジン

―――――――――――――――――――――――――――■◆■

こんにちは！ドリーム・アーツの廣瀬です。

涼しくなったかと思えば急に気温が上がったりと、なかなか不安定な時期が続いていますね。
私は週末に少し遠出をして紅葉を見に行きました🍁木々の葉っぱの色が変わっていく瞬間は、季節の変化を感じます。
皆さんが季節の変わり目を感じる瞬間はいつでしょうか？

今月もSmartDBお役立ち情報を配信していきます。

――本日のお役立ち情報――――――――――――――――――――――――――――

1. 10/18デジタルの民主化DAYを開催しました！
2. 10月のコミュニティサイト人気投稿ランキング
3. SmartDBのレビューを投稿してAmazonギフト券をゲット！

―――――――――――――――――――――――――――――――――――――――

DreamArts

SmartDB認定制度

SmartDB Certified Specialist

SmartDBによる
業務デジタル化の習熟度・スキルを
個人へ認定するプログラム

詳細はこちら▶ https://hibiki.dreamarts.co.jp/smartdb/scs/

すでに、多くの認定者が誕生！
デジタルの民主化を推進する企業が続々受験されています

JAPAN AIRLINES
Daiwa House 大和ハウスグループ
YAMADA HLDGS.
DLS Dai-ichi Life Group
三菱オートリース
SEKISUIKASEI
RITSUMEIKAN
sala サーラグループ
CARTA
HITOWA GROUP
OYO 応用地質株式会社
LIXIL トータルサービス
CAPCOM
アイダ設計
Tomorrow, Together KDDI

※弊社サイトより抜粋

【3月31日まで】コミュニティ参加者限定

ITreviewへのアンケート掲載でもれなくプレゼント！

Amazonギフト券 3,000円分

スマートデービー SmartDB × ITreview

キャンペーンコードを必ずご入力ください

AUQBMHOY

※「キャンペーン要項」を必ずご確認ください

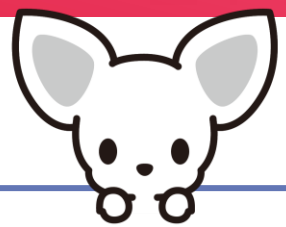
無くなり次第終了

Step.1 ITreviewに会員登録をしよう

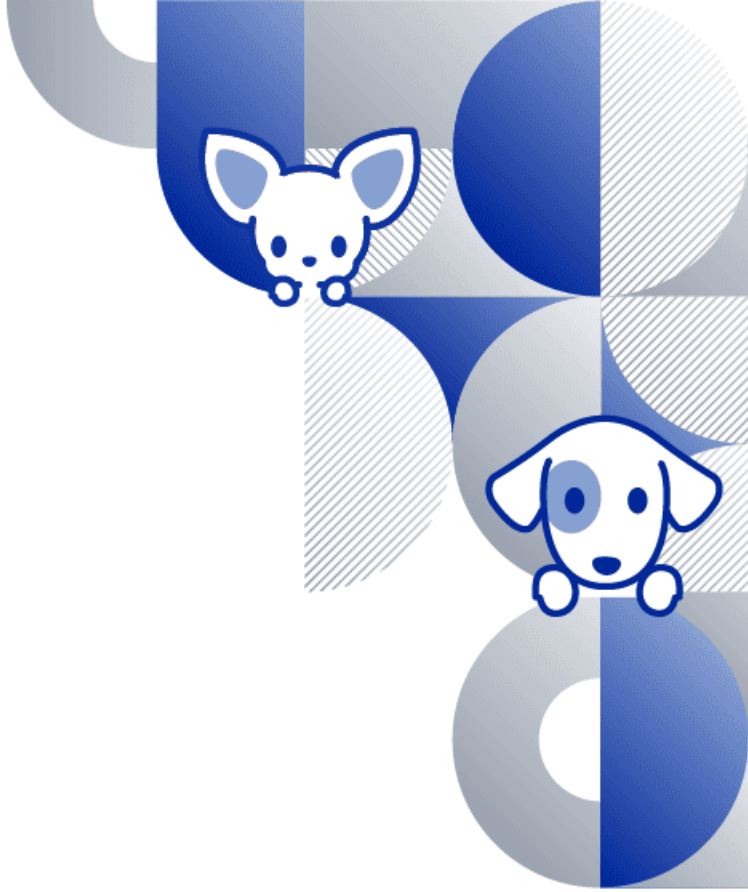
Step.2 レビュー・口コミを書こう

Step.3 掲載されると…

Step.4 Amazonギフト券GET!!



キャンペーン専用ページURL



DreamArts

https://www.dreamarts.co.jp/